Faster Quantum Chemistry Simulation on Fault-Tolerant Quantum Computers

Faster quantum chemistry simulation on fault-tolerant quantum computers

2012 New J. Phys. 14 115023

(http://iopscience.iop.org/1367-2630/14/11/115023)

New Journal of Physics
The open access journal for physics

Faster quantum chemistry simulation on fault-tolerant quantum computers

N Cody Jones[1,7], James D Whitfield[2,3,4], Peter L McMahon[1], Man-Hong Yung[2], Rodney Van Meter[5], Alán Aspuru-Guzik[2] and Yoshihisa Yamamoto[1,6]

[1] Edward L Ginzton Laboratory, Stanford University, Stanford, CA 94305-4088, USA
[2] Department of Chemistry and Chemical Biology, Harvard University, 12 Oxford Street, Cambridge, MA 02138, USA
[3] NEC Laboratories, America, 4 Independence Way, NJ 08540, USA
[4] Physics Department, Columbia University, 538 W 120th Street, New York, NY 10027, USA
[5] Faculty of Environment and Information Studies, Keio University, Japan
[6] National Institute of Informatics, 2-1-2 Hitotsubashi, Chiyoda-ku, Tokyo 101-8430, Japan
E-mail: ncodyjones@gmail.com

New Journal of Physics **14** (2012) 115023 (35pp)
Received 9 April 2012
Published 27 November 2012
Online at http://www.njp.org/
doi:10.1088/1367-2630/14/11/115023

Abstract. Quantum computers can in principle simulate quantum physics exponentially faster than their classical counterparts, but some technical hurdles remain. We propose methods which substantially improve the performance of a particular form of simulation, *ab initio* quantum chemistry, on fault-tolerant quantum computers; these methods generalize readily to other quantum simulation problems. Quantum teleportation plays a key role in these improvements and is used extensively as a computing resource. To improve execution time, we examine techniques for constructing arbitrary gates which perform substantially faster than circuits based on the conventional Solovay–Kitaev algorithm (Dawson and Nielsen 2006 *Quantum Inform. Comput.* **6** 81). For a given approximation error ϵ, arbitrary single-qubit gates can be

[7] Author to whom any correspondence should be addressed.

produced fault-tolerantly and using a restricted set of gates in time which is $O(\log \epsilon)$ or $O(\log \log \epsilon)$; with sufficient parallel preparation of ancillas, constant average depth is possible using a method we call programmable ancilla rotations. Moreover, we construct and analyze efficient implementations of first- and second-quantized simulation algorithms using the fault-tolerant arbitrary gates and other techniques, such as implementing various subroutines in constant time. A specific example we analyze is the ground-state energy calculation for lithium hydride.

Contents

1. Introduction

Simulating quantum physics from first principles is arguably one of the most important applications of a quantum computer—a problem intractable to solve in many cases, yet valuable to science [1]. The objective of quantum simulation is to model natural physical systems with Hamiltonians that permit a compact representation [2, 3]. Several different applications of a quantum physics simulator have been proposed, including: spin glasses [4] and lattices [5, 6]; Bardeen–Cooper–Schrieffer Hamiltonians [7, 8]; and quantum chemistry [9, 10]. More recently, Jordan *et al* [11] proposed a variant of this approach for simulating relativistic quantum field theories. In this investigation, we narrow our focus to quantum chemistry problems such as

calculating the eigenvalues of a molecular Hamiltonian [9, 12–14]. We aim to demonstrate constructively how quantum computers can simulate chemistry with an efficient use of resources by representing the molecular system with a first-principles Hamiltonian consisting of kinetic energy and Coulomb potential operators between electrons and nuclei. In doing so, we indicate how close the field of quantum information processing is to solving novel problems for less computational cost than a classical computer.

The motivation behind our study is that in order for computational physics on quantum computers to be useful as a scientific tool, it must have an efficient implementation. Often general algorithmic complexity such as 'polynomial time' is taken as a by-word for efficient, but we go deeper to show the substantial performance disparities between different polynomial-time algorithms, revealing which ones are significantly less costly in space and time resources than their peers. By introducing algorithmic improvements and making quantitative analysis of the resource costs, we show that simulating quantum chemistry is feasible in a practical execution time. An example problem we analyze is calculating the ground-state energy of lithium hydride (LiH) in 5.6 h on a hypothetical fault-tolerant quantum computer with an execution time per error-corrected gate of 1 ms. This stands in contrast to previous results based on slower methods [6], which would require 3.8 years to complete the same task on the same machine.

Quantum chemistry and band structure calculations account for up to 30% of the computation time used at supercomputer centers [15], and *ab initio* chemistry is one of the two physics-simulation applications which dominate the use of supercomputing resources (the other being fusion-energy research). The most-employed techniques include density functional theory and polynomially tractable approximate quantum chemistry methods [16]. Despite the success of these methods, for example, in simulating the dynamics of a small protein from first principles [17] or in predicting novel materials [18], they are still approximate, and much work is carried out in developing more accurate methods. Quantum simulators offer a fresh approach to quantum chemistry [19] as they are predicted to allow for the exact simulation (within a selected basis) of a chemical system in polynomial time. A quantum computer of a sufficient size, say 128 logical quantum bits [9, 20], would already outperform the best classical computers for *exact* chemical simulation. This would open the door to high-quality *ab initio* data for parameterizing force fields for molecular dynamics [21] or understanding complex chemical mechanisms such as soot formation [22], where a number of different chemical species must be compared. This tends to suggest that computational chemistry would be one of the first novel applications of universal quantum computers.

Several possible simulators have been proposed and studied [19, 23–26], but we focus on fault-tolerant circuit-model quantum simulation in this investigation [2, 6, 9, 10, 14, 20, 27, 28]. The reasons for these constraints are straightforward: quantum computers will probably be sensitive to noise and other hardware errors, thus requiring fault tolerance [29], and fault-tolerant quantum computing has been most successfully applied in the circuit model. Fault tolerance requires an additional overhead for the quantum computer; error correcting codes and the mechanisms they use to correct errors have been studied previously [29–31]. We focus here on another matter critical to simulation algorithms, which is making arbitrary fault-tolerant gates. Arbitrary quantum operations, such as a single-qubit rotation of arbitrary angle around the σ^z axis on the Bloch sphere, are typically constructed using a sequence of primitive error-corrected gates [30, 32, 33]. Quantum simulation depends sensitively on the execution time of arbitrary gates of this form, so one of the core contributions of this paper is to demonstrate efficient constructions for such gates, which would allow simulation of more complex systems

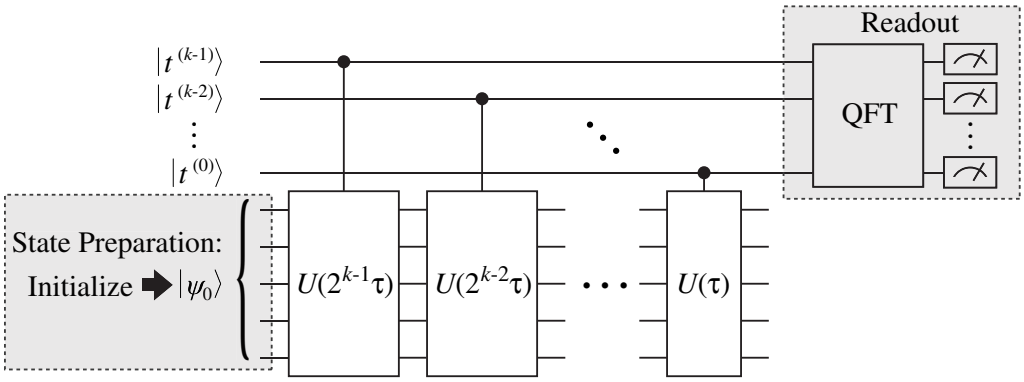

Figure 1. Schematic of a digital quantum simulation algorithm for energy eigenvalue estimation [2, 9]. The three main steps are state preparation, simulated evolution, and readout; this investigation focuses on the middle process. After preparing an initial state $|\psi_0\rangle$, the system is evolved in simulated time by solving the time-dependent Schrödinger equation. Note the system propagators $\mathcal{U}(2^x\delta t)$ are controlled by qubits in a register representing simulated time. A quantum Fourier transform (QFT) on the time register provides an estimate of an energy eigenvalue. The accuracy of the simulation depends on suppressing errors in both state preparation and simulated-time evolution, which is why fault tolerance is an important consideration for quantum simulation algorithms.

under a fixed-resource constraint. Many of the fast circuits we use can be understood as a form of quantum teleportation, and prior efforts have established the importance of teleportation and entanglement as information-processing resources [34–36].

A digital quantum simulation algorithm consists of three primary steps (figure 1): state preparation, simulated time evolution, and measurement readout. This paper focuses on the second step, evolving the system in simulated time, because this represents the core of the algorithm. Simulation of time evolution on a quantum computer is a sequence of quantum gates which closely approximates the evolution propagator $\mathcal{U}(t; t+\delta t) = \mathcal{T}\exp(-\frac{\mathrm{i}}{\hbar}\int_t^{t+\delta t}\mathcal{H}(\tau)\mathrm{d}\tau)$ of a desired Hamiltonian $\mathcal{H}$, where $\mathcal{T}$ is the usual time-ordering operator. In the case of a time-independent Hamiltonian, we have $\mathcal{U}(\delta t) = \exp(-\frac{\mathrm{i}}{\hbar}\mathcal{H}\delta t)$, as in figure 1. The increment δt is a single time step of simulation, and a simulation algorithm often requires many time steps, depending on the desired result (e.g. energy eigenvalue). State preparation and measurement readout are necessary steps which are not discussed here, but details can be found in references [3, 30, 37–40].

The quantum simulation problem we analyze is the ground-state energy calculation of LiH from first principles. This was called the 'chemist's workbench' and is an appropriate continuation of quantum computational applications of chemistry going beyond molecular hydrogen [10, 14, 41, 42]. For some of the selected methods, the quantum circuit is compact enough to be tractable for classical computation. Still, this example is useful for two reasons. Firstly, the LiH simulation preserves the features of more complicated chemical simulations while permitting a simple analysis that illustrates the improved methods we propose. Secondly, with quantum computers still in early stages of development, a compact problem such as LiH would be a convenient choice for experimental demonstrations of quantum simulation in the

near term. A larger, more complex simulation problem was studied in [43] using the methods analyzed here.

This paper provides constructive methods for simulating quantum chemistry efficiently using fault-tolerant quantum circuits. Section 2 describes how to construct quantum circuits for arbitrary phase rotations, which are essential to simulation. Section 3 develops a fault-tolerant simulation algorithm in second-quantized representation using phase rotations from the prior section; analysis of the computing resources required follows. Section 4 demonstrates how to construct an efficient chemistry simulation in first-quantized form, and total quantum resources are analyzed. Section 5 outlines how to determine the optimal simulation parameters for a given set of engineering constraints and performance objectives. The paper concludes by discussing the prospects for fault-tolerant quantum computers to solve novel simulation problems.

2. Fault-tolerant phase rotations

The algorithms which simulate chemistry on a circuit-model quantum computer require many phase rotations, accurate to high-precision. A single-qubit rotation gate in general form is

$$R_Z(\phi) = \mathrm{e}^{\mathrm{i}\frac{\phi}{2}}\mathrm{e}^{-\mathrm{i}\frac{\phi}{2}\sigma^z} = \begin{bmatrix} 1 & 0 \\ 0 & \mathrm{e}^{\mathrm{i}\phi} \end{bmatrix}, \tag{1}$$

where ϕ is arbitrary and σ^z is the Pauli spin operator. Additionally, any arbitrary single-qubit gate can be produced using three distinct phase rotations and two Hadamard gates [30]. Quantum error correction constrains the available operations to a finite set of fundamental gates, so the arbitrary rotations needed to simulate Hamiltonian evolution must be constructed with a circuit consisting of these fundamental gates. Phase rotations are needed at every time step of simulation, so the performance of the simulation algorithm depends on the computational complexity of these arbitrary-gate circuits. In this section, we discuss three different approaches for implementing arbitrary phase gates efficiently: *phase kickback* [44–46], which uses multi-qubit gates acting on an ancilla register; *gate approximation sequences*, such as those generated by the Solovay–Kitaev algorithm [30, 32] or by Fowler's algorithm [33], which are sequences of single-qubit gates; and *programmable ancilla rotations* (PARs), which compute ancillas in advance using one of the above methods to achieve very low circuit depth in the algorithm.

2.1. Phase kickback

Phase kickback [44, 45], also known as the Kitaev–Shen–Vyalyi algorithm [46], is an ancilla-based scheme that uses an addition circuit to impart a phase to a quantum register. Phase kickback relies on a resource state $|\gamma^{(k)}\rangle$ which can be defined by the inverse quantum Fourier transform (QFT) [30, 47, 48]:

$$|\gamma^{(k)}\rangle = U_{\mathrm{QFT}}^{\dagger}|k\rangle = \frac{1}{\sqrt{N}}\sum_{y=0}^{N-1}\mathrm{e}^{-2\pi\mathrm{i}ky/N}|y\rangle\,. \tag{2}$$

The register $|k\rangle$ contains n qubits prepared in the binary representation of k, an odd integer. The state $|\gamma^{(k)}\rangle$ is a uniform-weighted superposition state containing the ring of integers from 0 to $N-1$, where $N = 2^n$, and each computational basis state has a relative phase proportional to the equivalent binary value of that basis state. This ancilla register must be produced fault-tolerantly. Kitaev *et al* [46] provide a method to prepare $|\gamma^{(k)}\rangle$ using phase estimation such

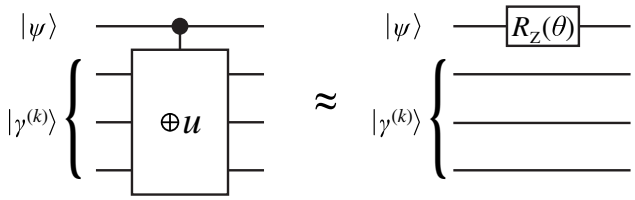

Figure 2. Controlled addition of the quantity u determined by equation (4) is approximately equivalent to an arbitrary phase rotation $R_Z(\phi)$, but the former uses only fault-tolerant gate primitives and ancillas. The operation $\oplus$ denotes unitary addition modulo 2^n, where n is the number of qubits in the $|\gamma^{(k)}\rangle$ register; for illustration, $n = 3$ in the circuits above.

that k is a random odd integer; hence our analysis does not assume a value for k. If necessary, appendix B provides a technique to convert any $|\gamma^{(k)}\rangle$ into $|\gamma^{(1)}\rangle$. The circuit complexity for creating $|\gamma^{(k)}\rangle$ is small, requiring perhaps a few thousand gates, so the cost of this initialization step is negligible compared to quantum algorithms we analyze later.

One could also view the $|\gamma^{(k)}\rangle$ state as a discretely sampled plane wave with wavenumber k. Consider then that $|\gamma^{(k)}\rangle$ is an eigenstate of the unitary operation $U_{\oplus u}|m\rangle = |m + u \pmod N\rangle$ for modular addition, so that

$$U_{\oplus u}|\gamma^{(k)}\rangle = \frac{1}{\sqrt{N}} \sum_{y=0}^{N-1} \mathrm{e}^{2\pi \mathrm{i} k(u-y)/N} |y\rangle = \mathrm{e}^{2\pi \mathrm{i} k u/N} |\gamma^{(k)}\rangle, \tag{3}$$

where $\oplus$ denotes addition modulo N and u is an integer. Moreover, the eigenvalue of modular addition on $|\gamma^{(k)}\rangle$ is a phase factor proportional to the number u added. Note that the addition operation $U_{\oplus u}$ is readily implemented with a fault-tolerant quantum circuit [49–53]. To determine the value of u in the addition circuit which approximates a phase rotation $R_Z(\phi)$, one solves the modular equation

$$ku \equiv \left\lfloor N \frac{\phi}{2\pi} \right\rceil \pmod N, \tag{4}$$

which always has a solution since k is odd and N is a power of 2 (k and N have no common factors). The operation $\lfloor x \rceil$ denotes rounding any real x to the nearest integer; any arbitrary rule for half-integer values suffices here. By proper selection of u, one can approximate any phase rotation to within a precision of $|\Delta\phi| \leqslant \frac{2\pi}{2^{n+1}}$ radians, where $|\Delta\phi| = \left[|\phi - \frac{2\pi}{N} ku| \pmod{2\pi}\right]$. We can now understand how the method received its name: since $|\gamma^{(k)}\rangle$ is an eigenstate of addition, when an integer u is added (using an addition circuit) to this register, a phase is 'kicked back.' This method is quite versatile, as several different types of phase gates are developed using phase kickback in this work.

Single-qubit phase rotations using phase kickback are constructed with a controlled addition circuit, as shown in figure 2. Intuitively, a phase is kicked back to the control qubit if it is in the $|1\rangle$ state, which is equivalent to the phase rotation in equation (1). The accuracy of the phase gate and the quantum resources required depend on the number of bits in the ancilla state $|\gamma^{(k)}\rangle$. After solving equation (4), the integer u is added to $|\gamma^{(k)}\rangle$ using a quantum adder controlled by the qubit which is the target of the phase rotation. There are various implementations of quantum adder circuits which have tradeoffs in performance between circuit

Table 1. Universal set of fault-tolerant gates in this investigation.

Symbol	Name	Matrix representation
X, Y, Z	Pauli gates	$\begin{bmatrix}0 & 1\\1 & 0\end{bmatrix}, \begin{bmatrix}0 & -\mathrm{i}\\\mathrm{i} & 0\end{bmatrix}, \begin{bmatrix}1 & 0\\0 & -1\end{bmatrix}$
H	Hadamard	$\frac{1}{\sqrt{2}}\begin{bmatrix}1 & 1\\1 & -1\end{bmatrix}$
S	$\pi/4$ phase gate	$\begin{bmatrix}1 & 0\\0 & \mathrm{i}\end{bmatrix}$
T	$\pi/8$ phase gate	$\begin{bmatrix}1 & 0\\0 & \mathrm{e}^{\mathrm{i}\pi/4}\end{bmatrix}$
CNOT	Controlled-NOT (two-qubit gate)	$\begin{bmatrix}1 & 0 & 0 & 0\\0 & 1 & 0 & 0\\0 & 0 & 0 & 1\\0 & 0 & 1 & 0\end{bmatrix}$

depth, circuit size, and difficulty of implementation [49–53]. Since $|\gamma^{(k)}\rangle$ is not altered by phase kickback, the number of such registers required for a quantum algorithm is equal to the maximum number of phase rotations which are computed in parallel at any point in the algorithm.

2.2. Gate approximation sequences

A gate approximation sequence uses a stream of fault-tolerant single-qubit gates to approximate an arbitrary phase rotation, such as that in equation (1). For context, a common set of fault-tolerant gates is listed in table 1. Such sequences must be calculated using a classical algorithm, and at least two options exist. The Solovay–Kitaev algorithm [30, 32] is perhaps the best known method for generating arbitrary quantum operations, so it will serve as a benchmark in our analysis. A subsequently derived alternative, Fowler's algorithm [33], offers shorter gate sequences for a given approximation accuracy, with some notable drawbacks in classical algorithmic complexity.

The efficiency of a gate approximation sequence is determined by the accuracy of approximation (i.e. how close the composite sequence is to the desired gate) as a function of resource costs. Both the Solovay–Kitaev and Fowler algorithms produce better approximations if one can afford more quantum gates; however, quantum resources are expensive, so we must implement finite-length sequences which produce a sufficiently good approximation. We adopt the distance measure in [33] to determine approximation accuracy:

$$\mathrm{dist}_d(U, V) = \sqrt{\frac{d - \left|\mathrm{tr}(U^\dagger V)\right|}{d}}, \tag{5}$$

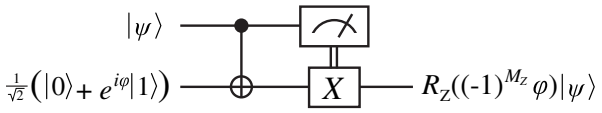

Figure 3. Probabilistic rotation using an ancilla qubit. The state of the top qubit is teleported to the bottom qubit with a phase rotation applied. The measurement is in the computational (Z) basis. The circuit enacts either $R_Z(\phi)$ or $R_Z(-\phi)$ with equal probability. The X gate is classically conditioned on the measurement result.

where d is the dimensionality of U and V (e.g. $d = 2$ for a single-qubit rotation). At the end of this section, we provide a quantitative analysis of resource costs to produce phase rotations. What is sufficient for the moment is to know that, if we denote the approximation error as $\epsilon = \mathrm{dist}_2(U, U_{\mathrm{approx}})$, the corresponding approximating sequence U_{approx} has asymptotic length $O(\mathrm{poly}(\log \epsilon))$, a result known as the Solovay–Kitaev theorem [30].

2.3. Programmable ancilla rotation

We introduce a third method for producing phase rotations, the PAR, which pre-computes ancillas before they are needed. Shifting the computing effort to a different point in the quantum circuit (assuming parallel computation) allows this method to achieve *constant average depth* in the algorithm for any desired accuracy of rotation, which can be as small as four quantum gates. The pre-calculated ancillas still require quantum circuits of similar complexity to the previously discussed methods, so this approach is best-suited to a quantum computer with many excess logical qubits available for parallel computing.

The PAR is based on a simple circuit which uses a single-qubit ancilla to make a phase rotation, which is a 'teleportation gate' [34, 35], as shown in figure 3. This circuit is probabilistic, so there is a 50% probability of enacting $R_Z(-\phi)$ instead of $R_Z(\phi)$; in such an event, we attempt the circuit again with angle 2ϕ, then 4ϕ if necessary, etc. This proceeds until the first observation of a positive angle rotation, in which case we have enacted a rotation $\phi_{\mathrm{total}} = 2^m\phi - \sum_{x=1}^{m-1} 2^x\phi = \phi$.

The circuit for the PAR is shown in figure 4. The programmed ancillas $|\omega^{(1)}\rangle = \frac{1}{\sqrt{2}}(|0\rangle + e^{i\phi}|1\rangle)$, $|\omega^{(2)}\rangle = \frac{1}{\sqrt{2}}(|0\rangle + e^{i(2\phi)}|1\rangle)$, etc are pre-computed using one of the methods above for a phase rotation. A very similar method was shown in [54], but we generalize here from $\phi = \frac{\pi}{2^k}$ to arbitrary rotation angles. In practice, phase kickback may be preferable for producing the pre-computed ancillas since reusing the same $|\gamma^{(k)}\rangle$ ancilla does not introduce additional errors into the circuit. The cascading series of probabilistic rotations continues until the desired rotation is produced or the programmed ancillas are exhausted. For practical reasons, one may only calculate a finite number of the PAR ancillas, and if all such rotations fail, then a deterministic rotation using phase kickback or a gate approximation sequence is applied. The probability of having to resort to this backstop is suppressed exponentially with the number of PAR ancillas pre-computed.

The average number of rounds of the circuit in figure 4 before a successful rotation is simply given by $\sum_{m=1}^{\infty} \frac{m}{2^m} = 2$. The X gate in each round can be performed with a Pauli frame [43, 55, 56], so counting measurement as a gate, the number of gates per round is 2, and the average number of gates per PAR is 4. With a finite number of pre-computed ancillas M,

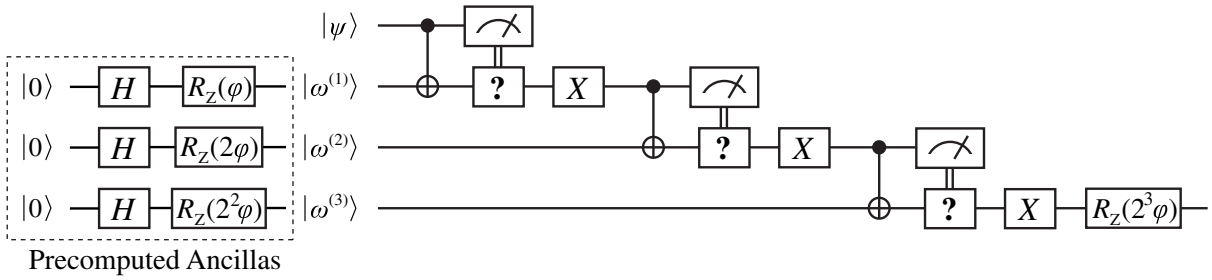

Figure 4. PAR circuit. The bulk of the computing effort is shifted to an earlier part of the circuit, when the ancillas are produced. The programmed ancillas are used in multiple rounds of the circuit in figure 3, each of which succeeds with 50% probability. The cascading circuit above terminates after the first success, as denoted by the '?' decision gates. The average number of rounds required is 2, so by pre-computing the ancillas, this method contributes very few additional gates to an algorithm's circuit depth.

there is a probability 2^{-M} of having to implement the considerably more expensive (in circuit depth) deterministic rotation. Nevertheless, if the computer supports the ability to calculate the programmed ancillas in advance, the PAR produces phase rotations that are orders of magnitude faster than other available methods, which also leads to faster execution of simulation algorithms.

2.4. Analysis of a single-qubit phase rotation

We begin our quantitative analysis by examining fault-tolerant single-qubit phase rotations. We construct rotations using phase kickback, the Solovay–Kitaev algorithm, Fowler's algorithm, and PARs. In each case, we determine the depth of the quantum circuit and the types of fault-tolerant gates required. The techniques developed here will be used in the more complicated phase rotations for the simulation algorithms in sections 3 and 4.

To assess the performance of quantum circuits, let us assume the following simplified quantum computing model. The hypothetical system uses fault-tolerant quantum error correction, so we presume the quantum gates are ideal. The quantum computer only has access to a limited set of 'fundamental' gates, which are summarized in table 1; this set of gates is typical for a fault-tolerant quantum computer [30, 43, 54, 57]. The available set of quantum gates is restricted by error-correction codes [30]. In essence, errors cannot be corrected for the continuous set of arbitrary quantum gates, just as in classical analogue computing. We allow full parallelism so that gates can be applied to all qubits simultaneously, as long as the two-qubit (CNOT) gates do not overlap. Because the fundamental gate set has a finite number of members, phase kickback or gate approximation sequences are required to produce approximations to arbitrary gates. We should note that each logical gate with error correction will require many more physical operations to be implemented [29, 43, 54], but we purposefully avoid these details so that our present analysis is independent of hardware and error correction models. To make a connection to future quantum hardware, separate investigations find that one error-corrected qubit requires about 1000 faulty physical qubits coupled into an encoded state [43, 54].

Therefore, when we state that 100 encoded qubits are required, this implies the quantum processor will contain 10^5 physical qubits.

When benchmarking the performance of a phase rotation, the important figures of merit are the quantum resources consumed to achieve a given accuracy of approximation. Using the distance measure in equation (5), the approximation error is quantified as

$$\epsilon = \mathrm{dist}_2\left(R_Z(\phi), U_{\mathrm{approx}}\right), \tag{6}$$

where U_{approx} is the circuit approximating $R_Z(\phi)$. Figure 5 reports two quantum resources for a single-qubit rotation: circuit depth, which is the minimum execution time in gates, and the total number of T gates required (see table 1). These calculations are explained in more detail in appendix A. T gates are significantly more expensive to prepare fault-tolerantly than other fundamental gates in many prominent error-correcting codes [30, 57], so they represent an important consideration for large-scale quantum computing [6, 43, 54]. It is apparent from figure 5 that Solovay–Kitaev sequences are substantially more expensive than their counterparts in both circuit depth and T gates. Fowler sequences are very compact and, in fact, optimal for an approximation sequence, but the classical algorithm to find them requires a calculation time that appears to grow exponentially faster than the other methods: $\epsilon \leqslant 10^{-2}$ requires minutes, $\epsilon \leqslant 10^{-3}$ requires about an hour, and $\epsilon \leqslant 10^{-4}$ requires about a day, for each rotation, on a modern workstation. For these reasons, phase kickback may be the method of choice when high-precision ($\epsilon \leqslant 10^{-6}$) rotations are required. Phase kickback requires quantum resources comparable to Fowler sequences, but the quantum circuit depends on adders, which are trivial to compile. The methods we analyze for producing fault-tolerant phase rotations are summarized in table 2.

3. Simulating chemistry in second-quantized representation

Simulation in the second-quantized form expresses the electronic Hamiltonian $\mathcal{H}$ in terms of the creation operators $a_p^\dagger$ and the wavefunction in terms of fermionic (or bosonic) modes $|p\rangle \equiv a_p^\dagger|0\rangle$ (i.e., occupation number representation). In chemistry, the single-electron molecular orbital picture has provided a practical method for approximating an N-electron wavefunction. Using second-quantized algorithms, basis sets in computational chemistry can be imported directly into quantum computational algorithms. For this reason, both theoretical [9, 12, 14] and experimental [10, 42] investigations in second-quantization have been performed.

Following the standard construction (see e.g. [19]), an arbitrary molecular Hamiltonian in second-quantized form can be expressed as

$$\mathcal{H} = \sum_{p,q} h_{pq} a_p^\dagger a_q + \frac{1}{2} \sum_{p,q,r,s} h_{pqrs} a_p^\dagger a_q^\dagger a_r a_s, \tag{7}$$

where $h_{pq} = \langle p|(\hat{T} + \hat{V}_N)|q\rangle$ are one-electron integrals ($\hat{T}$ is the kinetic energy operator, and $\hat{V}_N$ is the nuclear potential) and $h_{pqrs} = \langle pq|\hat{V}_e|rs\rangle$ represent the Coulomb potential interactions between electrons. All of the terms h_{pq}'s and h_{pqrs}'s are pre-computed numerically with classical computers, and the values are then used in the quantum computer to simulate the Hamiltonian evolution through the operators

$$U_{pq} = \mathrm{e}^{-\mathrm{i}h_{pq}(a_p^\dagger a_q + a_q^\dagger a_p)\delta t} \tag{8}$$

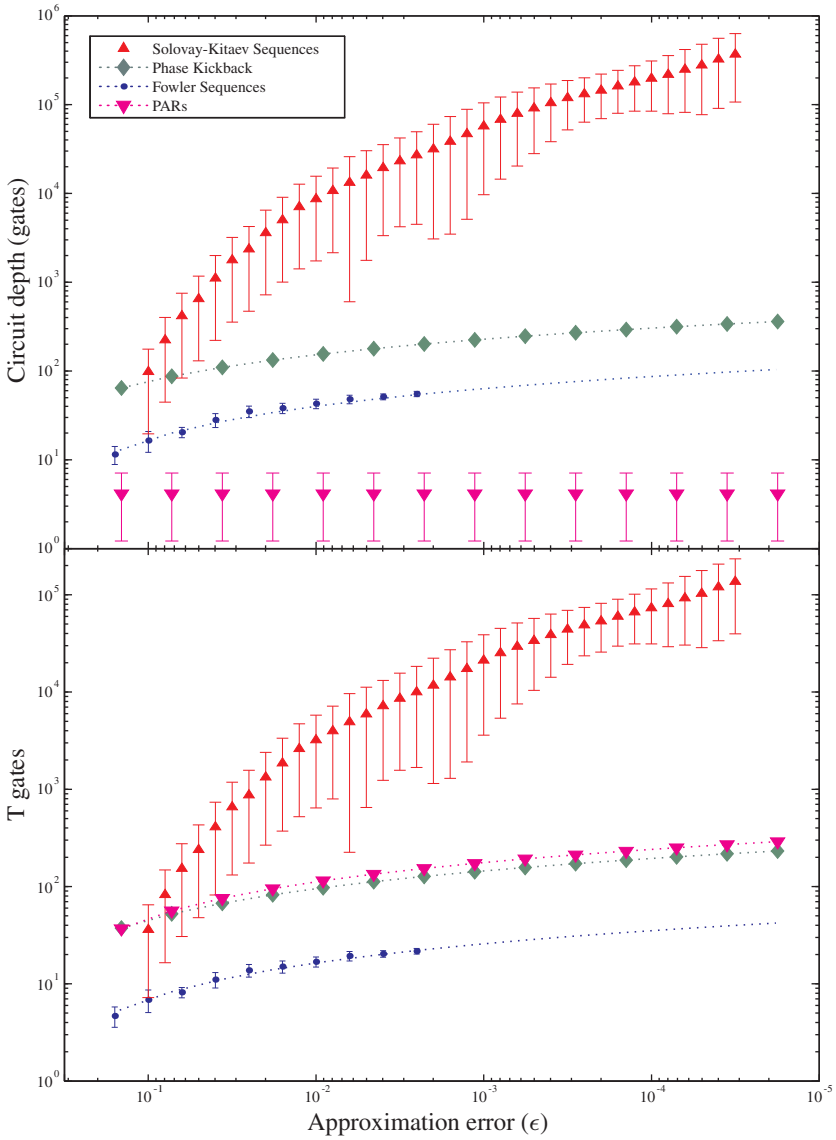

Figure 5. Quantum computing resources required to produce a fault-tolerant single-qubit phase rotation to accuracy $\epsilon = \mathrm{dist}_2\left(R_Z(\phi), U_{\mathrm{approx}}\right)$ using various methods. (Top) Circuit depth for single-qubit rotations. (Bottom) Number of T gates required for each rotation. There is variation in the resources required for Solovay–Kitaev sequences, Fowler sequences, and PARs; each point is the mean number of gates required, and where applicable, the bars show plus/minus one standard deviation. The Solovay–Kitaev data is averaged over 9534 random angles (ϕ), and the Fowler data is averaged over 98 random angles per point. Fowler sequences are numerically intensive to calculate, so curves fit to the data are shown for $\epsilon \leqslant 10^{-3}$: depth $= -24.9 \log_{10} \epsilon - 7.64$ and T gates $= -9.75 \log_{10} \epsilon - 2.81$. Phase kickback is implemented here with a ripple-carry adder [52]. PARs use six pre-computed ancillas. Solovay–Kitaev sequences were calculated using code written by Dawson and Nielsen [32]; Fowler sequences were calculated using code written by Fowler.

Table 2. Summary of methods for producing fault-tolerant phase rotations. The quantity ϵ is the accuracy of an approximate rotation, and is defined by equations (5) and (6).

Method	Description	Advantages	Disadvantages
Phase kickback	Approximates arbitrary phase rotation via controlled addition applied to the $\|\gamma^{(k)}\rangle$ ancilla register	Trivial to compile. Circuit depth is $O(\log \epsilon)$ or $O(\log \log \epsilon)$, depending on adder circuit	Requires a logical ancilla register consisting of $O(\log \epsilon)$ qubits. Resource costs are about 2–3× higher than Fowler sequences
Solovay–Kitaev sequence	Approximates arbitrary rotation with a sequence of fundamental gates. Depth is $O(\log^c \epsilon)$, with $c \approx 4$	Polynomial-time compiling algorithm. No logical ancilla states	Dramatically more expensive in quantum resources than alternatives
Fowler sequence	Approximates arbitrary rotation with a sequence of fundamental gates. Depth is $O(\log \epsilon)$	Minimal-depth sequences. No logical ancilla states	Sequence-determination algorithm has exponential complexity and becomes infeasible for high-accuracy rotations
PAR	Approximates arbitrary rotation with a probabilistic circuit using ancilla and measurement	Constant average depth (four gates) for any phase rotation	Requires logical ancillas which must be pre-computed

and

$$U_{pqrs} = \mathrm{e}^{-\mathrm{i}h_{pqrs}(a_p^\dagger a_q^\dagger a_r a_s + a_s{}^\dagger a_r{}^\dagger a_q a_p)\delta t}. \tag{9}$$

These operators are constructed with a Jordan–Wigner transform and an arbitrary controlled phase gate $\mathrm{CR}_Z(\phi)$ [14], as shown in figure 6. The Jordan–Wigner transform requires `H`, `S`, and `CNOT` gates, which are often readily available in fault-tolerant settings, so we focus first on the considerably more resource-intensive controlled phase rotations. We later show how to implement the Jordan–Wigner transform efficiently.

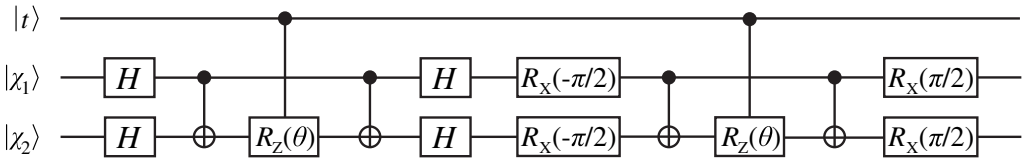

Figure 6. Excitation operator $e^{-ih_{12}(a_1^\dagger a_2 + a_2^\dagger a_1)\delta t}$ encoded into a quantum circuit [14]. Above, $\theta = h_{12}\delta t$. The gate $R_X(-\pi/2) = \mathrm{H} \cdot \mathrm{S}^\dagger \cdot \mathrm{H}$ is available from the set in table 1. In this example, the control qubit $|t\rangle$ is used for phase estimation, and the qubits $|\chi_1\rangle$ and $|\chi_2\rangle$ are basis functions (e.g. molecular orbitals). The controlled phase rotations $\mathrm{CR}_Z(\theta)$ must be approximated using circuits of available fault-tolerant gates.

3.1. Controlled phase rotations

As can be seen in figure 6, when U_{pq} or U_{pqrs} is implemented in a controlled operation (such as in energy eigenvalue estimation, see also figure 1), the core component of the circuit is a controlled phase rotation

$$\mathrm{CR}_Z(\phi) = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & e^{i\phi} \end{bmatrix}. \tag{10}$$

One way to implement the controlled rotation in equation (10) is to deconstruct the operation into CNOTs and single-qubit rotations [58], as shown in figure 7. Another method requires just one single-qubit rotation, as well as an ancilla $|0\rangle$, as shown in figure 8. Nielsen and Chuang [30] (p 182) provide a circuit decomposition for the Toffoli gate into gates in table 1. We use the circuit in figure 8 (requiring just one phase rotation) for the remainder of this paper, because the cost of one ancilla qubit is typically modest compared to a phase rotation. One can implement phase kickback, gate approximation sequences, or PARs to produce the single-qubit rotations, as in section 2.4. Additionally, the PAR construction can be modified to produce controlled rotations more directly. If the control qubit *only controls other circuits* between ancilla production and the time a controlled-PAR is needed, as is the case for phase estimation algorithms, one can create the ancillas (see figure 4) using controlled rotations with one of the above methods and produce a controlled-PAR with the same cascading circuit.

The different methods of producing a controlled phase rotation are analyzed in figure 9. We have excluded Solovay–Kitaev sequences, which permits a linearly scaled vertical axis, showing that each of these methods has execution time linear in $\log \epsilon$ or constant. As before, the values for Fowler sequences are extrapolated. We can see that Fowler sequences and phase kickback are separated by approximately a factor of 3 in execution time, and the choice between the two would be motivated by whether compiling the Fowler sequence is feasible or not. The PAR circuit requires one of the above methods to pre-compute ancillas.

3.2. Finite precision in pre-calculated integrals

The execution time of a second-quantized simulation algorithm is proportional to the number of integral terms h_{pq} and h_{pqrs}, as indicated by equations (7)–(9). We now consider how to speed up

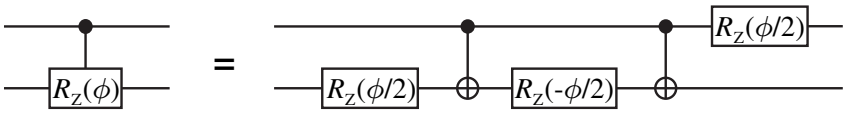

Figure 7. Decomposition of a controlled phase rotation into CNOTs and fault-tolerant single-qubit rotations. If the control qubit *only controls other circuits*, as in phase estimation algorithms, the third phase rotation commutes with the CNOTs. In such an event, the third single-qubit rotations from all decompositions of controlled rotations commute, and they can be combined into just one rotation prior to a non-commuting operation on this qubit (such as the QFT and measurement readout in figure 1). As a result, controlled rotations in phase estimation algorithms are effectively decomposed into two CNOTs and two single-qubit rotations with this circuit.

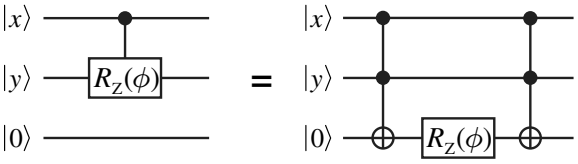

Figure 8. Controlled rotation $\mathrm{CR}_Z(\phi)$ (see equation (10)) between qubits $|x\rangle$ and $|y\rangle$ using two Toffoli gates, just one single-qubit rotation gate, and an ancilla $|0\rangle$. The ancilla qubit is conditionally set to $|1\rangle$ using a Toffoli gate, and a phase is imparted to this state with the rotation $R_Z(\phi)$. A final Toffoli gate returns the ancilla qubit to state $|0\rangle$.

the algorithm by omitting the integral terms that are negligibly small in magnitude. For a basis set consisting of M single-particle orbitals, the maximum number of integral terms is $O(M^4)$. In practice, however, the effort for evaluating these integrals often scales somewhere between $O(M^2)$ and $O(M^3)$ with modern implementations [59], because typically many integral terms may be neglected for being smaller in magnitude than a cutoff threshold. Consequently, the execution time of second-quantized simulation is determined by the number of pre-computed integrals of the form h_{pq} and h_{pqrs} of sufficiently large magnitude, as well as the efficiency of producing the corresponding arbitrary phase rotations in the quantum computer, such as $\mathrm{CR}_Z(h_{pq}\delta t)$ in the gate sequence for $\mathrm{e}^{-\mathrm{i}h_{pq}(a_p^\dagger(a)_q + a_q^\dagger a_p)\delta t}$ [14].

To illustrate how many integral terms are present in a typical chemical problem, we have calculated the integrals for a second-quantized simulation of LiH. We performed calculations in the minimal basis and in a triple-zeta basis, using the GAMESS quantum chemistry package [60, 61], at a bond distance of 1.63 Å, with an integral term cutoff of 10^{-10} in atomic units. We computed the number of integrals above cutoff using the STO-3G basis [62] containing 12 spin orbitals (6 spatial orbitals) and the TZVP basis [63] containing 40 spin orbitals (20 spatial orbitals). The cumulative number of integral terms as a function of cutoff in TZVP basis is plotted in figure 10. With the STO-3G basis, there were 231 non-zero molecular integrals, but only 99 of them were greater than 10^{-10} atomic units in magnitude. This is an order of magnitude below what is expected from $O(M^4)$ scaling. Considering the larger, more accurate basis set (TZVP), there were 22 155 non-zero integrals, but only 10 315 were greater than the

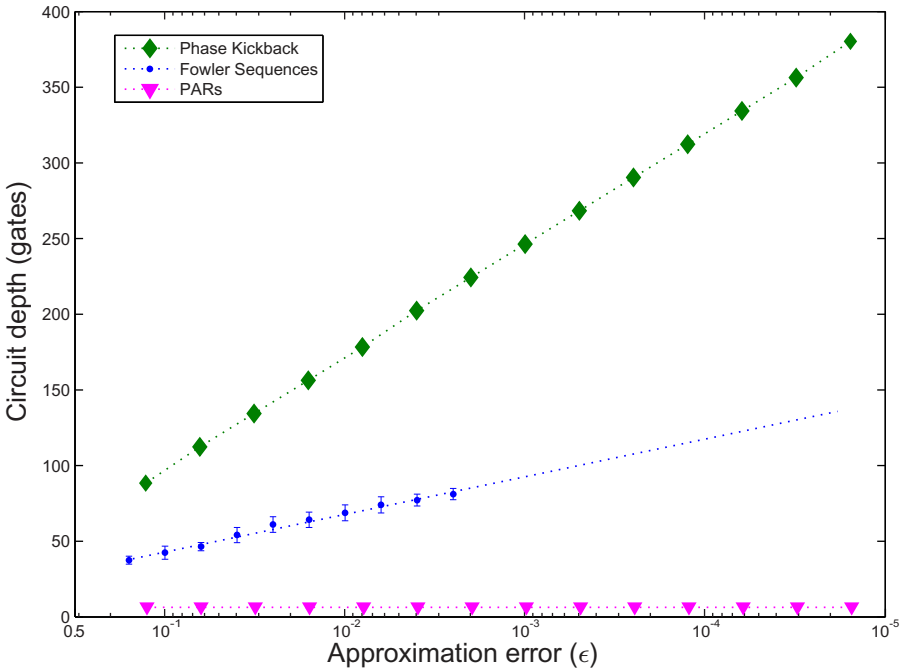

Figure 9. Circuit depth for controlled phase rotations using various methods. A desired controlled rotation $\mathrm{CR}_Z(\phi)$ is approximated with a fault-tolerant circuit U_{approx} with accuracy $\epsilon = \mathrm{dist}_4(U_{\mathrm{approx}}, \mathrm{CR}_Z(\phi))$ using the method in figure 8. Solovay–Kitaev sequences are omitted here to permit comparison of the more efficient schemes on a linear scale. The bars on Fowler sequence data indicate the standard deviation taken over 98 random-angle rotations. The controlled-PARs have a depth of four gates, on average, regardless of rotation accuracy. Phase kickback uses a ripple-carry adder since the addends have less than 16 bits [52]. If very high precision were desired, a carry-lookahead adder can achieve depth $O(\log\log\epsilon)$ at the expense of additional qubits and parallel circuits (more T gates) [53].

cutoff. Figure 10 shows that a higher cutoff, such as 10^{-4}, can further reduce the number of integrals in the TZVP basis implemented in the simulation. As a result, the effective number of integral terms the quantum computer must implement as phase rotations is nearly two orders of magnitude less than the asymptotic analysis would suggest. This is an example of the over-estimation of the resource costs that can occur when using asymptotic estimates. This technique becomes particularly relevant in large molecules since distant particles interact weakly, and in such an event, many of the associated integral terms may be negligibly small. Raising the cutoff threshold impacts the accuracy of the simulation, so one must attempt to balance the resource costs of simulation with the usefulness of the result.

3.3. Jordan–Wigner transform using teleportation

The second-quantized algorithm uses Jordan–Wigner transforms to implement operators such as $\mathrm{e}^{-\mathrm{i}h_{pq}(a_p^\dagger a_q + a_q^\dagger a_p)\delta t}$, and this section shows how to perform such transforms in constant time. As elaborated in [14], the circuits for Jordan–Wigner transforms often consist of ladders of CNOT

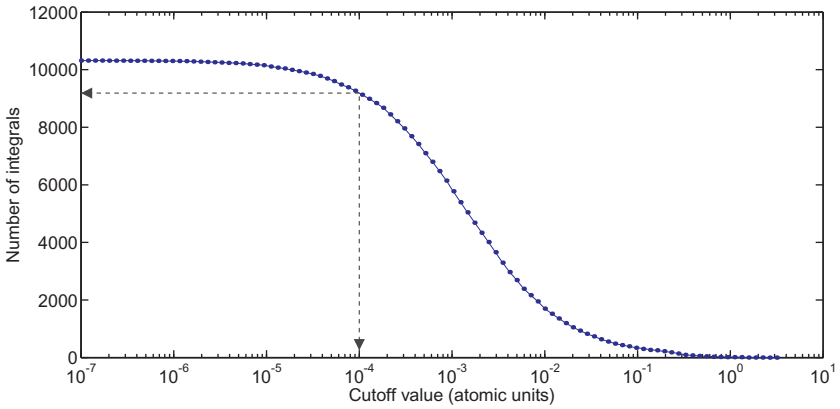

Figure 10. The number of integral terms implemented in a second-quantized simulation of LiH using a TZVP basis, as a function of cutoff threshold. Only integral terms with absolute value above the threshold are implemented in circuits, and the rest are neglected. As shown in the figure, a cutoff of 10^{-4} would require the algorithm to implement just over 9000 integral terms.

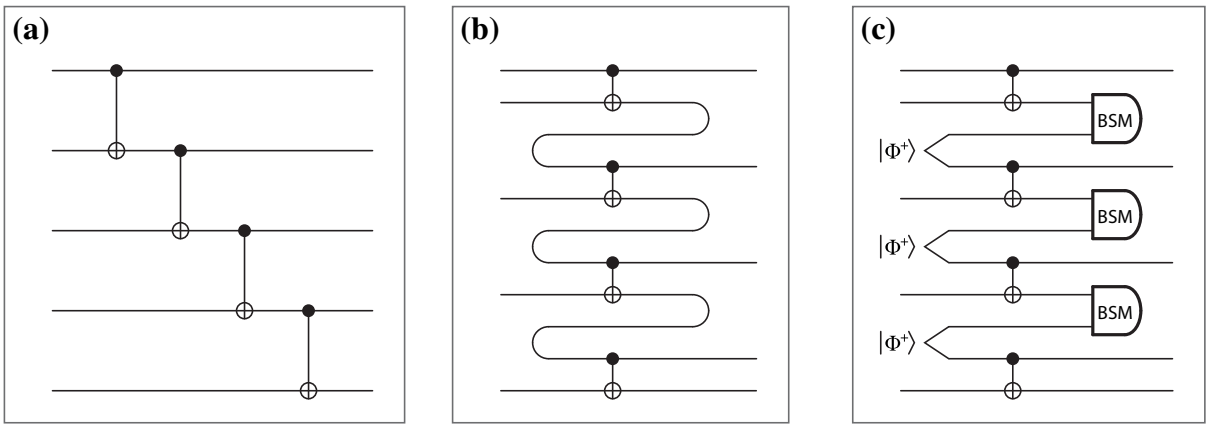

Figure 11. Rearrangement of the CNOT ladder common in Jordan–Wigner transforms using teleportation. (a) The original CNOT ladder requires an execution time that grows with the extent of the simulation in qubits. (b) A conceptual diagram of what teleportation accomplishes. The qubits 'move' backwards in time. (c) A valid quantum circuit that uses teleportation to move qubits in a manner which allows parallel computation of the CNOTs. The BSM is the Bell state measurement which teleports the qubits; the result of this measurement indicates the Pauli errors which are tracked by the Pauli frame [43]. The Bell state $|\Phi^+\rangle = \frac{1}{\sqrt{2}}(|00\rangle + |11\rangle)$ can be prepared from $|0\rangle$ ancillas using one H gate and one CNOT gate. Similarly, the BSM can be implemented using one H, one CNOT, and measurement of the two qubits in the computational basis.

gates, such as the one in figure 11(a). In a simulation with M basis states, these ladders can extend across the entire register of qubits corresponding to these basis states, which leads to the $O(M^5)$ asymptotic runtime quoted in [19] when there are at most $O(M^4)$ integral terms.

The CNOT ladder is a sparse network of Clifford gates, so we show how it may be implemented in constant time using teleportation [34, 35]. Figure 11(b) gives an intuitive picture for what will be accomplished. If the path of the qubits could be rearranged to somehow propagate backwards in time, the CNOT gates could be implemented simultaneously. Qubits cannot move backwards in time *per se*, but they can be moved arbitrarily using teleportation; notice how the conceptual (but unphysical) circuit in figure 11(b) is realized by a physical circuit in figure 11(c). Ancilla Bell states $|\Phi^+\rangle = \frac{1}{\sqrt{2}}(|00\rangle + |11\rangle)$ are used to teleport qubits in this rearranged CNOT ladder. Teleportation introduces a random Pauli error on the teleported qubit, but it is possible to track these errors and their propagation through CNOT gates using Pauli frames [43, 55, 56]. With this modification, it is possible to implement the Jordan–Wigner transform in constant time, which removes one of the bottlenecks to high-speed second-quantized simulation. This method could be adapted to implement other Clifford-group circuits in constant time, at the expense of requiring enough ancilla Bell states.

3.4. Resource analysis for ground-state energy simulation of lithium hydride

Using the hypothetical quantum computer from section 2.4, we examine the resources required to perform simulation in second-quantized form. Estimates of the number of qubits required for various instances of second-quantized chemical simulation have been reported previously [9, 19], so we focus instead on the execution time and effort to prepare fault-tolerant gates (here we consider the number of T gates). Figure 12 shows both the circuit depth and the number of T gates required to simulate LiH in the STO-3G basis as a function of rotation accuracy threshold $\epsilon_{\max}$, for 1023 simulated time steps. The precision in the readout is proportional the number of time steps simulated. The energy estimate in this simulation has 10 bits of precision, and in general, $2^n - 1$ steps are required for n bits of precision. If we assume that the duration of a single quantum gate is 1 ms (cf [43]), then the total execution time of the simulation ranges from $\sim$5.6 h using PARs to $\sim$3.8 years using Solovay–Kitaev rotations.

The number of T gates in figure 12 serves as an indication of the complexity demanded of the quantum computer. Although we do not delve into this matter, Jones *et al* [43] and Isailovic *et al* [54] discuss the importance (and difficulty) of producing these gates. What becomes apparent is that using PARs, while very fast, is also more expensive in the consumption of T gates than directly implementing Fowler sequences or phase kickback. Choosing between such approaches depends on the capabilities of the quantum computer, and we discuss this matter in more detail in section 5.

To provide an indication of how much execution time in second-quantized simulation is devoted to phase rotations, figure 13 shows the relative ratio of circuit depth devoted to implementing rotations versus all other gates for each of the methods considered when simulating LiH with rotation accuracy $\epsilon \leqslant 10^{-4}$. It is clear here that Solovay–Kitaev has such high circuit depth that it cannot be drawn to scale. We see also that Fowler and phase kickback sequences require execution times that are comparable, whereas PARs actually do not represent the majority of the circuit depth, unlike all of the prior methods. This is an encouraging result, because it shows that previous examinations that depended on Solovay–Kitaev sequences can be improved by orders of magnitude with more efficient phase rotations [6]. We do not consider Solovay–Kitaev sequences further in this investigation. The techniques for improving

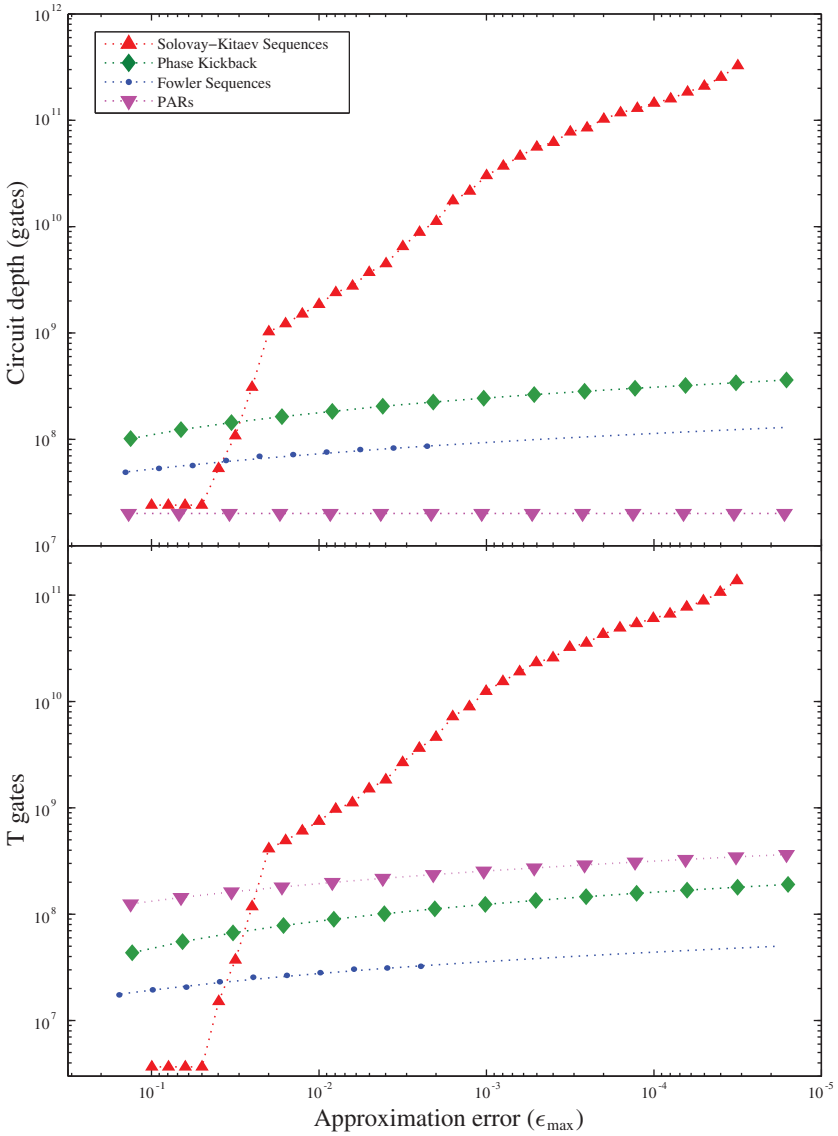

Figure 12. Total circuit depth and T gates for a second-quantized simulation of LiH using the STO-3G basis, calculated for different constructions of controlled rotations as a function of accuracy ϵ_{max}. For a given ϵ_{max}, every controlled rotation $CR_Z(\phi)$ in the algorithm is approximated with a fault-tolerant circuit U_{approx} with accuracy distance $\epsilon = dist_4(U_{approx}, CR_Z(\phi))$ such that $\epsilon \leqslant \epsilon_{max}$. An accuracy threshold $\epsilon_{max} \leqslant 10^{-4}$ is used in later analysis. This simulation implements all integral terms in the Hamiltonian (see equation (7)). (Top) Circuit depth using the gate set in table 1. In this plot, only the mean number of gates for PAR circuits is shown. (Bottom) T gates required for each method. The controlled-PAR ancillas are produced using controlled rotations constructed using Fowler sequences; six controlled-PAR ancillas are pre-computed for each rotation, and only mean values are plotted. The sudden jump in Solovay–Kitaev resource costs is because many controlled rotations in this algorithm have a small angle $\phi \approx 0$ that is approximated with identity gate at low precision, whereas the other methods are using a typical sequence length for arbitrary ϕ.

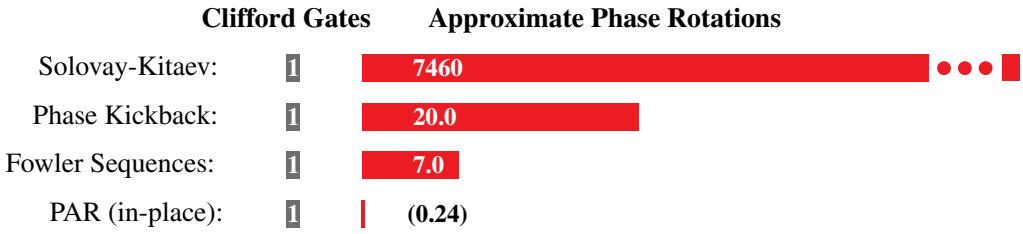

Figure 13. The relative amount of time (circuit depth) of a fault-tolerant, second-quantized simulation of LiH devoted to Clifford gates {X,Y,Z,H,S,CNOT} versus phase rotations that must be approximated. In this example, rotations are computed to an accuracy $\epsilon \leqslant 10^{-4}$. The relative circuit depth of rotations calculated by the Solovay–Kitaev algorithm is too large to be drawn to scale here. In the case of PAR, the ancillas must be pre-computed with a method such as Fowler sequences, but this can be carried out in parallel with other algorithm operations.

Table 3. Summary of methods for efficient second-quantized chemical simulation. The quantity M is the number of basis functions used in the representation of the chemical problem; larger basis sets produce more accurate results at the expense of greater circuit complexity.

Method	Description	Advantages	Disadvantages
Finite-precision cutoff in second-quantized integrals	Neglect to implement integral terms below a chosen cutoff in the algorithm execution	Second-quantized circuit complexity is reduced in both depth and the number of T gates	None if the cutoff threshold is below gate approximation accuracy
Jordan–Wigner transform using teleportation	Use a teleportation circuit to implement Jordan–Wigner transform in constant time	Second-quantized circuit depth reduces to at most $O(M^4)$ from $O(M^5)$	Teleportation circuit requires at most $3M-4$ qubits instead of M (only during Jordan–Wigner transform)

second-quantized simulation are summarized in table 3. The methods for determining resource costs are summarized in appendix A.

4. Simulating chemical structure and dynamics in first-quantized representation

The first-quantized simulation algorithm is in some ways more complex than the second-quantized algorithm, but for problems in chemistry larger than a handful of particles, it is computationally faster. A first-quantized simulation is essentially a finite-difference method for solving the Schrödinger equation. Configuration space is discretized into a Cartesian grid, and each particle (e.g. electron) has a wavefunction expressed in a quantum register that encodes a

probability amplitude at each coordinate on the grid. For example, let us imagine that we form a position-basis representation for a single electron on a $2^p \times 2^p \times 2^p$ grid, which requires only $3p$ qubits. Explicitly, the electronic wavefunction is represented as

$$|\psi_e\rangle = \sum_{x,y,z=0}^{2^p-1} c(x, y, z)\,|x\rangle\,|y\rangle\,|z\rangle = \sum_{\mathbf{r}} c(\mathbf{r})\,|\mathbf{r}\rangle\,, \tag{11}$$

where $c(x, y, z)$ is the complex probability amplitude for the electron to occupy the volume element centered at the position $\mathbf{r} \equiv (x, y, z)$. The rightmost part of equation (11) is shorthand that will be used throughout this section. The spin degree of freedom can easily be incorporated by including an extra qubit, and to describe a many-electron state, the wavefunction has to be properly anti-symmetrized [37, 64].

To simulate the evolution of a time-independent molecular Hamiltonian $\mathcal{H}$ for problems in quantum chemistry, we adopt the method given in [3, 20]. The complete Hamiltonian in first-quantized form can be expressed as the sum of the kinetic ($\hat{T}$) and potential ($\hat{V}$) operators

$$\mathcal{H} = \hat{T} + \hat{V} = -\sum_i \frac{\hbar^2 \nabla_i^2}{2m_i} + \frac{1}{2}\sum_{i\neq j} \frac{q_i q_j}{4\pi\epsilon_0 r_{ij}}, \tag{12}$$

where the indices i and j run over all particles (electrons and nuclei) of any given molecule. Here $r_{ij} \equiv |\mathbf{r}_i - \mathbf{r}_j|$ is the distance between particles i and j, which carry charges q_i and q_j respectively.

Let us outline how first-quantized simulation works before delving into details. The core of the algorithm is evolving the Hamiltonian in simulated time, achieved by applying the propagator $\mathcal{U}(t) = \exp(-i\mathcal{H}t)$ (setting $\hbar = 1$ and assuming $\mathcal{H}$ is time-independent), which solves the time-dependent Schrödinger equation [2]. This process is readily achieved using the split operator approximation, a form of Trotter–Suzuki decomposition [19, 27, 65, 66], where the kinetic and potential energy operators are simulated in alternating steps as

$$\mathcal{U}(t) = e^{-i\mathcal{H}t} \approx \left[e^{-i\hat{T}\delta t/2} e^{-i\hat{V}\delta t} e^{-i\hat{T}\delta t/2}\right]^{\frac{t}{\delta t}}. \tag{13}$$

The exponent $\frac{t}{\delta t}$ is the number of times the circuit corresponding to the expression in brackets is implemented, so it is always an integer. The operators $e^{-i\hat{V}\delta t}$ and $e^{-i\hat{T}\delta t}$ are diagonal in the position and momentum bases, respectively. One can switch the encoded configuration space representation between these two bases by applying the QFT to each spatial dimension of the wavefunction (cf equation (11)), which can be efficiently implemented in a quantum computer [48]. Kassal *et al* [20] show how to construct quantum circuits for operators $e^{-i\hat{V}\delta t}$ and $e^{-i\hat{T}\delta t}$, and in this section, we complement that work with analysis of fault-tolerant versions of these operators.

To make an algorithm fault-tolerant, its constituent operations must be decomposed into circuits of fault-tolerant primitive gates such as those in table 1. Consider the potential energy propagator $e^{-i\hat{V}\delta t}$ as an example. Given a b-particle wavefunction in the position basis as

$$\left|\psi_{1,2,\ldots,b}\right\rangle = \sum_{\mathbf{r}_1,\mathbf{r}_2,\ldots,\mathbf{r}_b} c(\mathbf{r}_1, \mathbf{r}_2, \ldots, \mathbf{r}_b)\,|\mathbf{r}_1\mathbf{r}_2\ldots\mathbf{r}_b\rangle\,, \tag{14}$$

where $c(\cdot)$ is the complex amplitude as a function of position in configuration space and subscripts correspond to particles in the system, one calculates the phase evolution of the

potential operator $e^{-i\hat{V}\delta t}$ in three steps, as follows:

$$\sum_{\mathbf{r}_1,\ldots,\mathbf{r}_b} c(\mathbf{r}_1,\ldots,\mathbf{r}_b)\,|\mathbf{r}_1\ldots\mathbf{r}_b\rangle\,|000\ldots\rangle$$

$$\longrightarrow \sum_{\mathbf{r}_1,\ldots,\mathbf{r}_b} c(\mathbf{r}_1,\ldots,\mathbf{r}_b)\,|\mathbf{r}_1\ldots\mathbf{r}_b\rangle\,|V(\mathbf{r}_1,\ldots,\mathbf{r}_b)\rangle \tag{15}$$

$$\longrightarrow \sum_{\mathbf{r}_1,\ldots,\mathbf{r}_b} e^{-iV(\mathbf{r}_1,\ldots,\mathbf{r}_b)\delta t} c(\mathbf{r}_1,\ldots,\mathbf{r}_b)\,|\mathbf{r}_1\ldots\mathbf{r}_b\rangle\,|V(\mathbf{r}_1,\ldots,\mathbf{r}_b)\rangle \tag{16}$$

$$\longrightarrow \sum_{\mathbf{r}_1,\ldots,\mathbf{r}_b} e^{-iV(\mathbf{r}_1,\ldots,\mathbf{r}_b)\delta t} c(\mathbf{r}_1,\ldots,\mathbf{r}_b)\,|\mathbf{r}_1\ldots\mathbf{r}_b\rangle\,|000\ldots\rangle\,. \tag{17}$$

Firstly, equation (15) calculates the potential energy as a function of position coordinates [20] (note that $\hat{V}$ is diagonal in this basis) and stores the result in a quantum register $|V(\mathbf{r}_1, \mathbf{r}_2, \ldots, \mathbf{r}_b)\rangle$ to some finite precision. Appendix C describes how to implement this quantum circuit for molecular Hamiltonians. Secondly, equation (16) uses the $|V(\mathbf{r}_1, \mathbf{r}_2, \ldots, \mathbf{r}_b)\rangle$ register in a 'quantum variable' phase rotation that imparts a phase to each grid point of the wavefunction in position basis proportional to the potential energy at those coordinates. This section discusses how to implement the quantum variable rotation (QVR) using fault-tolerant quantum circuits. Finally, the quantum circuit from the first step is reversed in equation (17) to reset the $|V(\mathbf{r}_1, \mathbf{r}_2, \ldots, \mathbf{r}_b)\rangle$ register to $|000\ldots\rangle$, also known as 'uncomputation' [30]. The sequence of these three steps is equivalent to the operation $e^{-i\hat{V}\delta t}\,|\psi\rangle$.

The kinetic energy propagator $e^{-i\hat{T}\delta t}$ is calculated similarly in three steps, with the second also being a QVR. This operator is diagonal in momentum basis, so we transform the representation of the system wavefunction from position basis $\{x, y, z\}$ to momentum basis $\{k_x, k_y, k_z\}$ by applying a QFT along each spatial dimension of the encoding in equation (11). This form permits efficient calculation of the kinetic energy operator [20], which is described in appendix C.

4.1. Quantum variable rotation

The phase rotation subroutine in the first-quantized simulation algorithm imparts a quantum phase to each binary-encoded phase state in a superposition $|\theta\rangle = \sum_j c_j \left|\phi_j\right\rangle$ stored in a quantum register (c_j's are arbitrary complex amplitudes). Formally, it is the transformation

$$\sum_j c_j|\phi_j\rangle \longrightarrow \sum_j e^{2\pi i\xi\phi_j} c_j|\phi_j\rangle, \tag{18}$$

which generalizes the operation in equation (16) using ξ, which is a scaling factor that varies with implementation, as explained below and in appendix C. Each $0 \leqslant \phi_j < 1$ is a finite binary representation of a rotation on the unit circle encoded in a quantum register (the angle, in radians, divided by 2π). Equation (18) is the QVR, which is essential to first-quantized simulation. We show how to implement this phase rotation subroutine using phase rotations from previous sections, as well as a new construction based on phase kickback. At the end of the section, we analyze the resource costs of these methods.

To produce a QVR, various circuit manipulations are possible. The first is to simply apply a single-qubit rotation to each qubit in register $|\theta\rangle$, as shown in figure 14. Each individual rotation could be created using the techniques in section 2. Since a t-bit QVR requires t

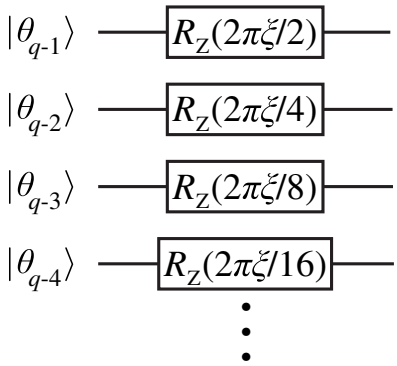

Figure 14. QVR decomposed into single-qubit rotations applied to each qubit in the $|\theta\rangle$ register consisting of q qubits (see equation (18)). $|\theta_{q-1}\rangle$ refers to the most significant bit in the register $|\theta\rangle$, etc.

separate bitwise rotations, we require that each rotation has accuracy ϵ/t to achieve accuracy ϵ in the QVR, where we have used the fact that the distance measure in equation (5) obeys the triangle inequality [33]. If the QVR is controlled by another qubit (e.g. if the propagator is controlled by a 'simulated time' qubit as in figure 1), then the gates in figure 14 are replaced with controlled rotations from section 3.1. In either case, one must know the quantity ξ in advance to compile these gates; typically, ξ is a product of physical constants and simulation parameters, as explained in appendix C.

The QVR can also be produced in a more elegant manner using phase kickback. Rather than applying bitwise gates to the $|\theta\rangle$ register, we instead use the entire register in a modified version of the phase kickback procedure. Firstly, we require a binary approximation to ξ, denoted $[\xi]$. Secondly, we define some quantities that describe this quantum circuit. Let m denote the number of significant bits in $[\xi]$, minus the number of trailing zeros. Define $w = \lfloor \log_2[\xi] \rfloor$, or in other words, w is the largest integer such that $2^w \leqslant [\xi]$. Denote $p = (m-1) - w$, which is how many bits we must shift $[\xi]$ up to produce an odd integer (if $p < 0$, we shift down). Following equation (18) and the preceding text, let q be the number of qubits in $|\theta\rangle$. Define integers $k_{[\xi]} = (2^p)[\xi]$ and $u_\phi = (2^q)\phi$ for some arbitrary $\phi \in [0, 1)$ represented using q bits. Thirdly, we construct a phase kickback ancilla register $|\gamma^{(k_{[\xi]})}\rangle$ of size $n = p + q$ qubits, using techniques in appendix B. Finally, we perform phase kickback with an addition circuit between registers $|\theta\rangle$ and $|\gamma^{(k_{[\xi]})}\rangle$ (in-place addition applied to $|\gamma^{(k_{[\xi]})}\rangle$), except this time the $|\theta\rangle$ register is shifted in one of two ways, as shown in figure 15. If $p \geqslant 0$, then the $|\theta\rangle$ register is shifted down by p qubits, and the $|\theta\rangle$ register is padded with p logical zeros at the most-significant side of the adder input (figure 15(a)). If $p < 0$, then $|\theta\rangle$ is shifted up by $|p|$ qubits, so that the $|p|$ most-significant bits of $|\theta\rangle$ are not used in the adder (figure 15(b)). If $n \leqslant 0$, then all rotations are identity and no QVR circuit is constructed.

We now confirm that this procedure produces the intended QVR. Using equation (3), we see that the above procedure will implement a phase rotation of

$$\sum_j c_j \left|\phi_j\right\rangle \longrightarrow \sum_j \mathrm{e}^{2\pi \mathrm{i} k_{[\xi]} u_{\phi_j} / 2^{p+q}} c_j \left|\phi_j\right\rangle. \tag{19}$$

Since $k_{[\xi]} = (2^p)[\xi]$ and $u_\phi = (2^q)\phi$, this is the same as

$$\sum_j c_j \left|\phi_j\right\rangle \longrightarrow \sum_j \mathrm{e}^{2\pi \mathrm{i}[\xi]\phi_j} c_j \left|\phi_j\right\rangle, \tag{20}$$

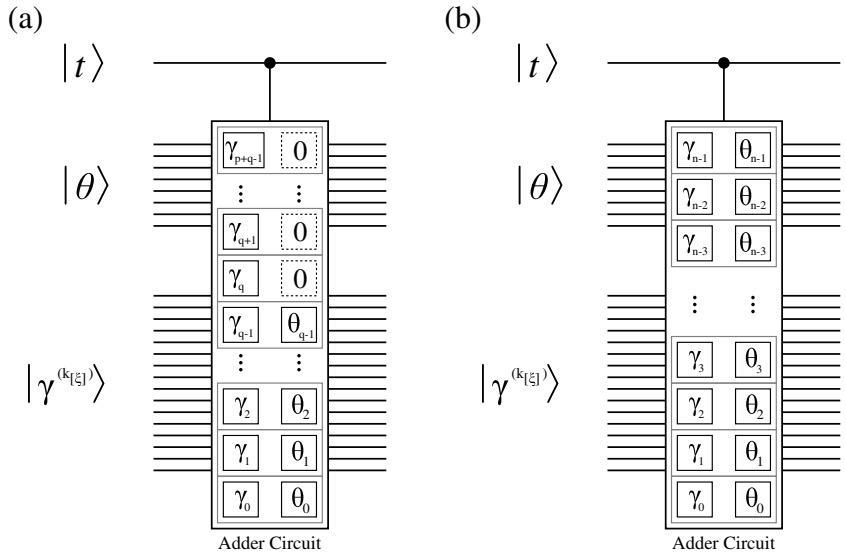

Figure 15. QVR using phase kickback. This circuit implements the operation in equation (18) with scaling factor $[\xi]$, which has been 'programmed' into the phase kickback register $|\gamma^{(k_{[\xi]})}\rangle$ (see appendix B). A control qubit $|t\rangle$ is included for illustration. This figure shows how the bits in the adder are aligned for different cases. (a) The register $|\theta\rangle$ is shifted down p bits since $p \geqslant 0$. θ_0 is the least-significant bit in the $|\theta\rangle$ register, etc. The input qubits above $|\theta\rangle$ are logical zeros. (b) The register $|\theta\rangle$ is shifted up $|p|$ bits since $p < 0$. In this case, the $|p|$ most-significant bits of $|\theta\rangle$ are not used in the adder.

which is equivalent to equation (18) using our finite representation for ξ. As before, if we require a controlled-QVR, then the adder can be controlled by an external qubit, which is the configuration shown in figure 15. This 'quantum variable' phase kickback uses substantially fewer T gates than the bitwise approach, as shown in figure 16, while having comparable circuit depth. Moreover, since there is only one phase rotation instead of many, it does not have to be as accurate as the individual rotations in figure 14 must be to achieve the same total accuracy in the QVR.

It may seem inefficient to produce a different phase kickback register for each QVR operation, but three properties of the first-quantized simulation algorithm make this approach efficient. Firstly, there are only a polynomial number of such operations: for b particles, there are b QVRs in the kinetic energy operator and $\frac{1}{2}b(b-1)$ QVRs in the potential operator. Secondly, many of these QVRs have the same scaling factor ξ, so a phase kickback register can be reused many times without modification. For example, the scaling factor in the kinetic energy operator is the same for all electrons and for all nuclei with the same mass. Third, the $|\gamma^{(k_{[\xi]})}\rangle$ registers can be calculated independently of other operations in the algorithm, so the impact of this process on circuit depth is minimal.

This phase kickback QVR has interesting applications to other useful quantum circuits. It can be used to make a fault-tolerant QFT; one replaces each block of controlled rotations with a controlled-QVR. As before, this approach uses substantially fewer T gates than an equivalent circuit where each controlled rotation in the QFT is implemented individually with techniques

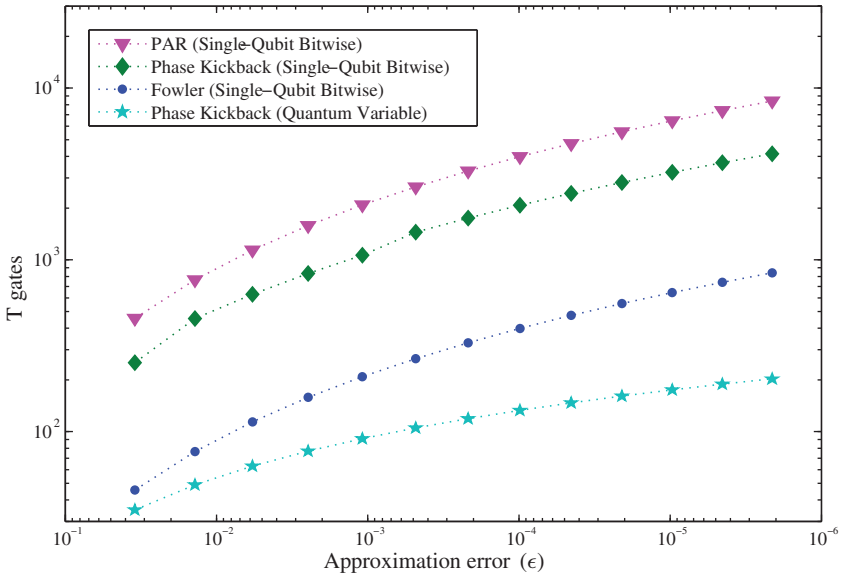

Figure 16. Number of T gates required to produce a QVR with various methods, assuming $\xi = 1$ and number of significant figures is chosen to satisfy the approximation error ϵ. The special-purpose 'quantum variable' phase kickback clearly requires the least circuit effort, and the asymptotic scaling of T gates is linear in $\log \epsilon$ for this approach and super-quadratic for the others. The circuit depth for Fowler or phase kickback approaches is equivalent to the comparable single-qubit rotation; however, the PAR must succeed across all individual rotations for this circuit to succeed, so the mean circuit depth increases slightly. In the above, ten rounds of PAR ancilla are pre-computed for each single-qubit rotation in the QVR.

in section 3.1, and the same methods can be applied to an approximate QFT [67] by simply truncating the size of the $|\gamma^{(1)}\rangle$ register. The phase kickback QVR can also be used to efficiently produce ancillas for PAR if the particular rotation $R_Z(\phi)$ is required frequently, which can have applications to second-quantized simulation. If we denote the state $|+\rangle = \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle)$, then an input state of $|+\rangle\,|+\rangle\,|+\rangle \ldots$ will be transformed using QVR (with appropriate ξ) into the set of ancillas for PAR, but requiring only one addition circuit for the entire set instead of a phase kickback addition or Fowler sequence for each ancilla qubit, which can be seen by comparing figure 14 with the ancilla preparation in figure 4. Creating the necessary $|\gamma^{(k_{[\xi]})}\rangle$ for this process is costly, so there is a net gain only if a certain rotation angle ϕ is required often.

4.2. Improved parallelism in potential energy operator

The majority of the circuit effort in first-quantized simulation is devoted to calculating the potential energy [20]. We introduce here a technique to substantially speed up the calculation of the potential energy operator $\hat{V}$, which is simply the sum of the Coulomb interactions $\hat{V}_{ij} = \frac{q_i q_j}{4\pi\epsilon_0 r_{ij}}$ between all pairwise combinations of the electrons and nuclei. Note that this operator is a function of the positions $\mathbf{r}_i$ of the system particles only, so it is diagonal in the

position basis $|\mathbf{r}_1\mathbf{r}_2\ldots\mathbf{r}_b\rangle$. This fact means that all terms $\hat{V}_{ij}$ commute with each other, so they may be calculated in any order. Moreover, there are many sets of the $\hat{V}_{ij}$ operators that are disjoint, which means that each particle in the system is acted on by just one operator in the set. Using this observation, for example, we may calculate the Coulomb interaction $\hat{V}_{12}$ between particles 1 and 2 at the same time as $\hat{V}_{34}$ between particles 3 and 4, and so on. In general, for a system of b particles, there are $\frac{1}{2}b(b-1)$ pairwise interactions, and we can perform $\lfloor\frac{b}{2}\rfloor$ pairs in parallel, which means that a potential energy operator with $O(b^2)$ terms can be calculated in $O(b)$ time. This parallelism can increase the speed of simulation significantly since evaluation of the potential energy dominates resource costs [43].

The potential operator calculation can be further parallelized to achieve $O(\log b)$ or $O(1)$ (constant) circuit depth. Exploiting the fact that all $\hat{V}_{ij}$ are diagonal in position basis (and hence commute), we use transversal CNOT gates to copy the data in the position-basis particle wavefunction onto multiple empty quantum registers. For a single particle, this process is

$$\left(\sum_{x,y,z=0}^{2^p-1} c(x,y,z)\,|x\rangle\,|y\rangle\,|z\rangle\right)|000\ldots\rangle\,|000\ldots\rangle\ldots$$
$$\rightarrow \sum_{x,y,z=0}^{2^p-1} c(x,y,z)\,(|x\rangle\,|y\rangle\,|z\rangle)\,(|x\rangle\,|y\rangle\,|z\rangle)\,(|x\rangle\,|y\rangle\,|z\rangle)\ldots. \tag{21}$$

For b particles, the copy operation is performed $b-2$ times (for $b-1$ total copies), which can be fanned out using a binary tree with depth $\lceil\log_2(b-1)\rceil$; constant depth can be achieved in some quantum computer architectures which support one-control/many-target CNOTs [43, 57] or in general architectures using a teleportation circuit similar to those described in section 3.3. This approach is similar to that employed in [47] to produce a parallel circuit for the QFT. The system wavefunction is now expanded to the state

$$\left|\psi_{\text{expand}}\right\rangle = \sum_{\mathbf{r}_1,\ldots,\mathbf{r}_b} c(\mathbf{r}_1,\ldots,\mathbf{r}_b)\,(|\mathbf{r}_1\rangle)^{\otimes(b-1)}\ldots(|\mathbf{r}_b\rangle)^{\otimes(b-1)}, \tag{22}$$

which requires $O(b^2)$ memory space. Note that this process is not cloning—the position-basis particle registers are still entangled to one another. With multiple accessible copies of each particle's position-basis information, the particles are matched in all $b(b-1)$ possible pairings, and the potential energy operator applied to each pairing in parallel, which can be accomplished in constant time, but still requires $O(b^2)$ circuit effort. After each of the potential energy operators $\hat{V}_{ij}$ kicks back a phase, the excess copies of each particle wavefunction are uncomputed by reversing the tree of CNOTs above. The preceding example demonstrates that it is possible to calculate $\hat{V}$ in time which is sub-linear in the number of particles, even if each $\hat{V}_{ij}$ is treated as a black box operator. In practice, more efficient circuits can be produced by generating the internal 'workspace' registers of $\hat{V}$ in parallel, rather than making copies of the input registers $\sum_{\mathbf{r}_1,\ldots,\mathbf{r}_b} c(\mathbf{r}_1,\ldots,\mathbf{r}_b)\,|\mathbf{r}_1\ldots\mathbf{r}_b\rangle$ (see appendix C).

4.3. Resource analysis for first-quantized molecular simulations

The advantage of using the first-quantized approach is that the approximation errors of the simulation are systematically improvable by increasing the spatial precision of the wavefunction

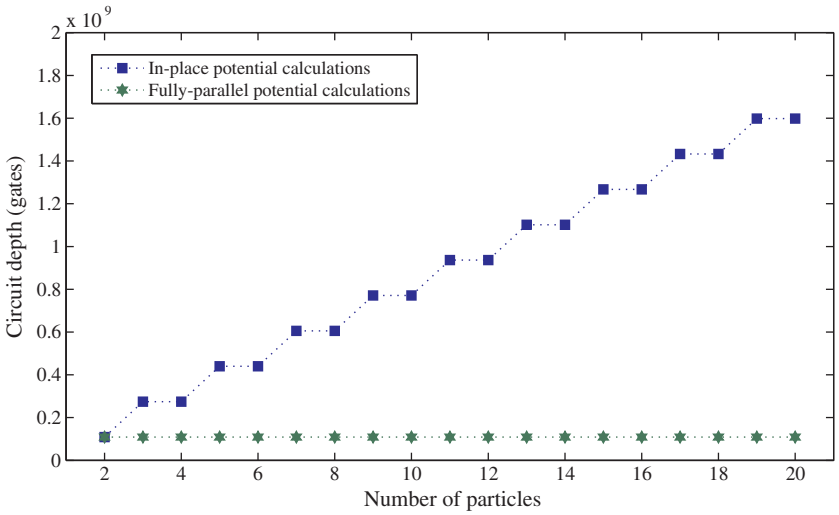

Figure 17. Circuit depth for two instances of first-quantized simulation. The in-place calculation of potential energy computes each pairwise Coulomb interaction in sets of non-overlapping particle pairs, and both the depth and number of qubits required increase linearly with the number of particles. The fully parallel calculation creates many copies of the wavefunction to permit the potential energy to be determined in constant time, at the expense of requiring substantially more application qubits (quadratic in the number of particles). In both cases, the wavefunction precision along any spatial dimension is ten qubits, and the simulation uses 1023 time steps for 10 bits of precision, or ∼3 significant figures.

and the temporal precision of the time steps. However, calculating kinetic and potential energy interactions requires quantum arithmetic circuits and phase rotations, which together require substantial resources in terms of fault-tolerant gates and qubits. Figure 17 shows two versions of first-quantized simulation using the techniques for parallel calculation of potential energy from the previous section. Although constant-depth evaluation of the Hamiltonian is possible, it requires a significantly larger quantum computer to achieve the parallel calculations, so this implementation is probably best suited to large-scale quantum computers.

Examining figure 17, note that the circuit depth at six particles (e.g. LiH) is comparable to that of the equivalent PAR-based second-quantized simulation in figure 12 while requiring many more qubits, indicating that first-quantized simulation is more appropriate for larger molecules than LiH, since the circuit depth for first-quantized simulation is asymptotically less than second-quantized as particle number is increased [19]. Moreover, these calculations have assumed that the spatial precision is ten qubits for any molecules with 2–20 particles. As the size of the molecule increases, the number of qubits for each dimension of the encoded wavefunction will have to increase as the molecule itself is spatially larger. One may also choose to increase spatial resolution to achieve a higher-precision simulation. Each of the methods we propose for improving first-quantized simulation are summarized in table 4. Appendix A explains how the resources were calculated.

Table 4. Summary of methods for efficient first-quantized chemical simulation. The quantity b is the number of particles in the chemical problem, which influences algorithm resource costs.

Method	Description	Advantages	Disadvantages
QVR	Use phase kickback to apply a fault-tolerant phase rotation to each element in a superposition, proportional to the binary-encoded value of that element	Reduces complexity of first-quantized simulation. Circuit depth is essentially the same as single-qubit phase kickback, but the QVR requires substantially fewer T gates than the method in figure 14	Not the minimal depth achievable, such as with PARs
Parallel evaluation of potential energy terms	Reduce potential operator circuit depth using parallel computation	Shorter circuit depth than calculating all $\frac{1}{2}b(b-1)$ terms individually	Concurrent computation requires more T gates simultaneously
Teleportation circuit expansion for potential operator	Use a teleportation circuit to 'control-copy' position-basis wavefunction in constant time	Potential operator can be evaluated in a time which is independent of problem size	Circuit size in qubits increases to $O(b^2)$ from $O(b)$

5. Comparing simulation methods

The prior sections illustrate that there exist numerous ways to simulate a molecular Hamiltonian, including choices between encoded representation in a quantum computer and the way fault-tolerant rotations are prepared. The final result one desires to know is, which method is best? Determining an optimal approach is subjective to the quantum computing resources available, so in this section we describe how to make such a decision.

To visually compare different implementations of a simulation algorithm, we plot the *efficient frontier* for each method in a plane defined by machine size (qubits) on the x-axis and execution time (circuit depth) on the y-axis. The efficient frontier is the set of all points (size, depth) such that for each achievable machine size, the (achievable) depth is minimized, and vice versa. As an example, figure 18 shows the efficient frontiers of various implementations of a LiH simulation.

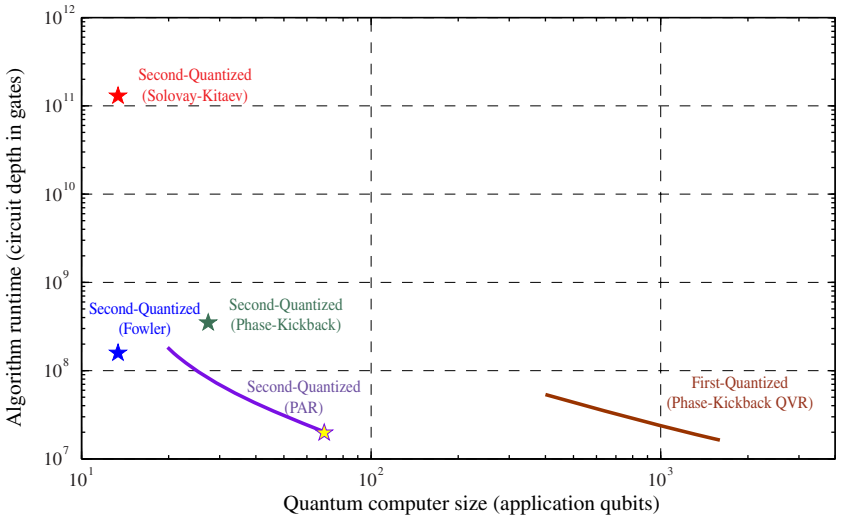

Figure 18. The efficient frontiers for various implementations of simulating LiH ground state energy on a quantum computer. Each star data point corresponds to the equivalent method in figure 12, at rotation accuracy $\epsilon_{\max} \leqslant 10^{-4}$; similarly, first-quantized simulations use QVRs with the same accuracy. The PAR frontier (purple) and first-quantized frontier (brown) have adjustable parameters that reduce circuit depth through parallel computation at the expense of increased system size (application qubits). For example, the PAR-based algorithm only achieves the circuit depth shown in figure 12 when the system has 68 qubits, which is the yellow star here.

To determine the optimal implementation, one specifies a cost function $g(x, y)$, which associates with any point (x, y) a 'cost' to implement simulation using these parameters. For example, cost could be the estimated engineering challenge to produce a quantum computer of size x qubits combined with a penalty for the execution time of y gates, which is a measure of performance. Minimizing the cost function along each efficient frontier gives the optimal set of parameters for that particular method, and minimizing over all efficient frontiers gives the best implementation that is known to be achievable.

For the various implementations for a LiH simulation in figure 18, it seems likely that one would choose between the compact algorithm with Fowler gate sequences or the faster version with PAR sequences, which requires additional qubits to compute the necessary ancillas. First-quantized can potentially deliver the fastest execution time here, but for this problem the number of qubits required is substantially greater. Still, first-quantized gains an appreciable performance advantage if the number of particles is increased or if one moves to simulating time-varying dynamics [19].

Naturally, future algorithm advancements could produce new frontiers that are more desirable for a given cost function. In general, one would like to make such comparisons, which can inform design decisions for quantum hardware, with full consideration of the cost to implement error correction, produce non-Clifford group gates (e.g. T gates), and so forth. Future work building on this investigation and prior efforts, such as [6, 43, 54, 68], should include such comprehensive system analysis.

6. Conclusions

This paper examines the methods required to simulate chemistry on a fault-tolerant quantum computer. A crucial operation in these algorithms is the production of phase rotations, and several approaches—phase kickback, gate approximation sequences, PARs, and QVRs—are analyzed. First, it should be clear that sequences generated by the Solovay–Kitaev algorithm are not nearly as efficient as the alternatives, phase kickback and Fowler sequences. Fowler sequences are the shortest for a fault-tolerant single-qubit rotation, but the classical computing effort required to determine such sequences becomes intractable for high-precision (e.g. $\epsilon < 10^{-6}$) rotations. Recently, Bocharov and Svore have presented an alternative algorithm for finding depth-optimal gate sequences which might feasibly provide high-precision rotations [69]. Phase kickback is a versatile technique that produces rotations comparable to Fowler's algorithm in resource usage, with the former having circuit depth $O(\log \epsilon)$ or $O(\log \log \epsilon)$ gates and requiring $O(\log \epsilon)$ T gates. Furthermore, the underlying circuit for phase kickback is an adder, which can be determined using efficient classical algorithms (unlike Fowler's algorithm), and phase kickback can be extended more readily to QVRs. The PAR allows the quantum algorithm to achieve exceptionally low-circuit-depth rotations, at the expense of computing ancillas in advance (which is less efficient in terms of T gates). Finally, the QVR is particularly useful for first-quantized simulation. The relative merits of the methods for producing phase rotations are compared in table 2.

This investigation also examined two variants of the simulation algorithm, second-quantized and first-quantized, whose primary difference is the way wavefunctions are encoded and operated upon. Generally speaking, second-quantized is a more compact representation, requiring fewer qubits, but it requires asymptotically longer execution times than first-quantized, measured in circuit depth, as the problem size increases in terms of independent particles to simulate. Our results provide a more nuanced way to compare these methods by explicitly considering the possible ways to make the algorithms compatible with fault-tolerant quantum computing and the resulting resource costs incurred. We have also introduced several improvements to the simulation algorithms. In the second-quantized approach, one can neglect some of the integral terms smaller in magnitude than a cutoff threshold, implement the Jordan–Wigner transform in constant time, and use PARs to substantially reduce circuit depth, at the expense of requiring parallel production of the pre-computed PAR ancillas. In first-quantized, we demonstrated how to produce QVRs with arbitrary scaling factor, as well as how to parallelize the calculation of the potential energy to time linear in system size (without increase in qubits) or to constant time (requiring a number of qubits that grows quadratically instead of linearly with the number of particles simulated). The methods we present for efficient chemical simulation on quantum computers are summarized in tables 3 and 4.

Although we have focused on simulating quantum chemistry, these methods can be extended to simulating other Hamiltonians on quantum computers, such as spin lattice models [4], lattice gas automata [70] and lattice gauge theories [71], or quantum chaos theories [72]. Moreover, the fault-tolerant rotations could find application in other quantum algorithms, including any which require a Fourier transform. This investigation provides a flexible set of methods for making simulation algorithms practically realizable on fault-tolerant quantum computers.

Acknowledgments

The authors would like to thank Kevin Obenland for suggesting improvements to quantum circuits; Paul Pham for providing assistance with Solovay–Kitaev code; Aram Harrow and Isaac Chuang for helpful discussions on Solovay–Kitaev; and Austin Fowler for providing code for minimum-length approximation sequences. This work was supported by the National Science Foundation CCF-0829694, the University of Tokyo Special Coordination Funds for Promoting Science and Technology, NICT, and the Japan Society for the Promotion of Science (JSPS) through its 'Funding Program for World-Leading Innovative R&D on Science and Technology (FIRST Program).' NCJ was supported by the National Science Foundation Graduate Fellowship. JDW acknowledges support from National Science Foundation award number 1017244. MHY and AAG acknowledge support from the Air Force Office of Scientific Research, award number FA8721-05-C-0002. AAG acknowledges support from the Alfred P Sloan Foundation, and the Camille and Henry Dreyfus Foundation.

Appendix A. Methods for calculating resources

We briefly describe how the resources for quantum circuits were calculated. Beginning with single-qubit phase rotations, the Solovay–Kitaev algorithm is described in [32], while the Fowler algorithm is described in [33]. Gate sequences for rotations were generated using these algorithms, and the total circuit depth and number of `T` gates were recorded. Phase kickback implements an adder circuit, and we select the CDKM adder (named for the authors of [52]). For reference, an n-qubit addition for phase kickback requires the n-qubit ancilla state $|\gamma^{(k)}\rangle$ and n ancilla qubits. The latter ancilla qubits (representing an input to the adder) are initialized to the binary representation of the rotation angle, conditioned on the qubit to which the rotation is applied, which is accomplished using $2n$ `CNOT` gates that can be implemented in constant time using the methods in section 3.3. The depth of the adder is $2n - 3$ Toffoli gates and 5 `CNOT` gates. Our implementation of the Toffoli gate uses 11 time steps, but recent work shows many variations are possible [69]. The depth of our n-bit adder-based phase-kickback rotation is $22n - 26$, which includes the parallel `CNOT` gates for setting the ancilla addend. Each Toffoli gate requires 7 `T` gates, so the CDKM adder has $14n - 21$ `T` gates in total.

Controlled phase rotations are built using single-qubit rotations. The Fowler and Solovay–Kitaev sequences require effectively two single-qubit rotations (see section 3.1); the rotations are inverses of each other, so the circuits for the two rotations are mirror images, and the total resources (depth and `T` gates) are doubled. Phase kickback requires only the slightest modification to become a controlled rotation. The n ancilla states are initialized to rotation angle conditioned on both the control and target qubit for this rotation. This requires two Toffolis (one at the start and one at the end) and $2n - 2$ additional `CNOT` gates (which can be implemented in a parallel fashion). The second-quantized algorithm is implemented using the methods described in [14] and each of these types for controlled rotations. The depth and `T` gates are given by the number of integral terms times the controlled phase rotation resource requirements, plus circuits for each of the necessary Jordan–Wigner transforms.

The first-quantized simulation algorithm is implemented using arithmetic circuits and QFTs. Resource analysis for exact and approximate QFTs is given in [30, 47, 48]. Detailed explanation of the arithmetic operations for this algorithm is given in [20] and appendix C. Each arithmetic circuit is decomposed into CDKM adders. The phase rotations in the kinetic

operators, potential operators, and QFT are implemented using phase-kickback QVRs, each of which requires a single CDKM adder.

Appendix B. Transforming the phase kickback register

In some situations it is useful to change the k-value in $|\gamma^{(k)}\rangle$, the phase kickback ancilla register (see equation (2)). Without control over k, the QVR in section 4.1 would require solving equation (4) in a quantum circuit; this step would in turn require a multiplication operation, which can be expensive in terms of quantum gates. We deviate here from [46] and propose a simple way to avoid the expensive operations associated with modular multiplication. The specific ancilla state $|\gamma^{(1)}\rangle$ does not require an additional circuit to solve equation (4), so we create this state explicitly using a simple transform $|\gamma^{(k)}\rangle \rightarrow |\gamma^{(1)}\rangle$. We begin by factoring the $|\gamma^{(k)}\rangle$ register into individual qubits (note that all such states are separable, i.e. not entangled):

$$\begin{aligned} |\gamma^{(k)}\rangle &= \frac{1}{\sqrt{N}} \sum_{y=0}^{N-1} \mathrm{e}^{-2\pi \mathrm{i} k y/N} |y\rangle \\ &= \frac{1}{\sqrt{N}} \left(|0\rangle + \mathrm{e}^{-2\pi \mathrm{i} k/2} |1\rangle\right) \otimes \left(|0\rangle + \mathrm{e}^{-2\pi \mathrm{i} k/4} |1\rangle\right) \otimes \cdots \\ &\quad \otimes \left(|0\rangle + \mathrm{e}^{-2\pi \mathrm{i} k/2^n} |1\rangle\right). \end{aligned} \tag{B.1}$$

We convert this state into $|\gamma^{(1)}\rangle$ with a series of single-qubit phase rotations using the controlled addition circuit from figure 2. Since k is odd, the first bit of our ancilla register is always $\frac{1}{\sqrt{2}}(|0\rangle - |1\rangle)$. The next bit must be rotated by the phase gate $R_Z(\pi(k-1))$, which is either identity or Z, depending on k. In general, the corrective gate applied to the mth bit is the phase rotation $R_Z(2\pi \frac{k-1}{2^{m-1}})$, which may be produced using the preceding $m-1$ bits of the ancilla register and phase kickback. By iterating through all qubits in the register, we complete the transformation with circuit depth $O(n^2)$ gates or less, depending on the type of adder used in phase kickback. This procedure can be generalized to any transformation $|\gamma^{(k)}\rangle \rightarrow |\gamma^{(l)}\rangle$ for *odd* integers $1 \leqslant k, l < 2^n$, where n is the number of bits in the phase kickback register.

Appendix C. Quantum circuits for potential and kinetic energy operators in first-quantized molecular Hamiltonians

First-quantized molecular simulation represents the simulated system wavefunction on a Cartesian grid, and the Hamiltonian is calculated with digital arithmetic acting on this coordinate space. Similar methods were discussed in the supplementary material of [20], but we update this analysis for the QVR introduced in this work. The potential energy operator is diagonal in position basis, and is the sum of Coulomb interactions between electrons and nuclei in the system $\hat{V} = \frac{1}{2}\sum_{i \neq j} \hat{V}_{ij}$, where

$$\hat{V}_{ij} = \frac{q_i q_j}{4\pi\varepsilon_0} \left(\frac{1}{|\mathbf{r_i} - \mathbf{r_j}|}\right) \tag{C.1}$$

and q_j is the charge of particle j. The prefactor on the rhs of equation (C.1) is a constant for any given pair of particles, and we can later encode this scaling factor into the QVR. What remains is to calculate $\frac{1}{|\mathbf{r_i} - \mathbf{r_j}|}$ over the position-encoded wavefunction. Each position register

can be decomposed in Cartesian components $|\mathbf{r}\rangle = |x\rangle\,|y\rangle\,|z\rangle$, so for a pair of particles we calculate

$$\left|{r_{ij}}^2\right\rangle = \left|\left(x_i - x_j\right)^2 + \left(y_i - y_j\right)^2 + \left(z_i - z_j\right)^2\right\rangle. \tag{C.2}$$

The required multiplication operations can be implemented using quantum adder circuits. Next, the quantity $|\frac{1}{r_{ij}}\rangle$ is calculated using the Newton–Raphson method with the iterative equation

$$a_{n+1} = \tfrac{1}{2}a_n\left(3 - a_n{}^2 {r_{ij}}^2\right). \tag{C.3}$$

With a suitably chosen initial value a_0, equation (C.3) converges within five iterations at 32-bit arithmetic, and typically less precision is required for simulation. The register $|\frac{1}{r_{ij}}\rangle$ is used in a QVR with scaling factor $\xi = \frac{q_i q_j \delta t}{8\pi^2 \varepsilon_0 \hbar}$ from above, where δt is the time step of this simulated evolution and an additional factor $1/2\pi$ comes from equation (18). Note that each component of $|\frac{1}{r_{ij}}\rangle$ is entangled to a position-basis component of the system wavefunction, so the QVR effectively kicks back a phase to the wavefunction. Each of the steps prior to the QVR is uncomputed, and the net effect of this sequence of operations is to implement the potential energy propagator $\mathrm{e}^{-\mathrm{i}\hbar^{-1}\hat{V}_{ij}\delta t}$, as in equations (15)–(17).

The kinetic energy operator is calculated using a similar approach as the potential energy. The kinetic energy is the sum of individual kinetic energy operators on each particle: $\hat{T} = \sum_j \hat{T}_j$, where

$$\hat{T}_j = \frac{{\hat{p}_j}^2}{2m_j} = \frac{\hbar^2 |\mathbf{k}_j|^2}{2m_j}. \tag{C.4}$$

The quantity m_j is the mass and $\mathbf{k}_j = \mathbf{p}_j/\hbar$ is the non-relativistic wavevector corresponding to particle j. By performing a QFT along each spatial dimension of the wavefunction, the system representation is transformed from position basis to momentum basis: $\{x, y, z\} \to \{k_x, k_y, k_z\}$. This form permits immediate calculation of magnitude squared of the wavevector:

$$\left||\mathbf{k}|^2\right\rangle = \left|{k_x}^2 + {k_y}^2 + {k_z}^2\right\rangle. \tag{C.5}$$

The $\left||\mathbf{k}|^2\right\rangle$ register is used in a QVR with scaling factor $\xi = \frac{\hbar\delta t}{4\pi m_j}$. Afterwards, the intermediate registers used in the calculation of $||\mathbf{k}|^2\rangle$ are uncomputed, and the end result is the operator $\mathrm{e}^{-\mathrm{i}\hbar^{-1}\hat{T}_j\delta t}$.

References

[1] Feynman R 1982 Simulating physics with computers *Int. J. Theor. Phys.* **21** 467
[2] Lloyd S 1996 Universal quantum simulators *Science* **273** 1073–8
[3] Zalka C 1998 Simulating quantum systems on a quantum computer *Proc. R. Soc. Lond.* A **454** 313–22
[4] Lidar D A and Biham O 1997 Simulating Ising spin glasses on a quantum computer *Phys. Rev.* E **56** 3661–81
[5] Master C P, Yamaguchi F and Yamamoto Y 2003 Efficiency of free-energy calculations of spin lattices by spectral quantum algorithms *Phys. Rev.* A **67** 032311
[6] Clark C R, Metodi T S, Gasster S D and Brown K R 2009 Resource requirements for fault-tolerant quantum simulation: the ground state of the transverse Ising model *Phys. Rev.* A **79** 062314
[7] Wu L-A, Byrd M S and Lidar D A 2002 Polynomial-time simulation of pairing models on a quantum computer *Phys. Rev. Lett.* **89** 057904
[8] Brown K R, Clark R J and Chuang I L 2006 Limitations of quantum simulation examined by simulating a pairing hamiltonian using nuclear magnetic resonance *Phys. Rev. Lett.* **97** 050504

[9] Aspuru-Guzik A, Dutoi A D, Love P J and Head-Gordon M 2005 Simulated quantum computation of molecular energies *Science* **309** 1704–7
[10] Lanyon B P *et al* 2010 Towards quantum chemistry on a quantum computer *Nature Chem.* **2** 106–11
[11] Jordan S P, Lee K S M and Preskill J 2012 Quantum algorithms for quantum field theories *Science* **336** 1130–3
[12] Wang H, Kais S, Aspuru-Guzik A and Hoffmann M R 2008 Quantum algorithm for obtaining the energy spectrum of molecular systems *Phys. Chem. Chem. Phys.* **10** 5388–93
[13] Veis L and Pittner J 2010 Quantum computing applied to calculations of molecular energies: CH_2 benchmark *J. Chem. Phys.* **133** 194106
[14] Whitfield J D, Biamonte J and Aspuru-Guzik A 2011 Simulation of electronic structure Hamiltonians using quantum computers *Mol. Phys.* **109** 735–50
[15] US Department of Energy 2010 *National Energy Research Scientific Computing Center: 2010 Annual Report, Technical Report*, Ernest Orlando Lawrence Berkeley National Laboratory, Berkeley, CA
[16] Head-Gordon M and Artacho E 2008 Chemistry on the computer *Phys. Today* **61** 58–63
[17] Ufimtsev I S, Luehr N and Martinez T J 2011 Charge transfer and polarization in solvated proteins from *ab initio* molecular dynamics *J. Phys. Chem. Lett.* **2** 1789–93
[18] Sokolov A N *et al* 2011 From computational discovery to experimental characterization of a high hole mobility organic crystal *Nature Commun.* **2** 437
[19] Kassal I, Whitfield J D, Perdomo-Ortiz A, Yung M-H and Aspuru-Guzik A 2011 Simulating chemistry using quantum computers *Annu. Rev. Phys. Chem.* **62** 185–207
[20] Kassal I, Jordan S P, Love P J, Mohseni M and Aspuru-Guzik A 2008 Polynomial-time quantum algorithm for the simulation of chemical dynamics *Proc. Natl Acad. Sci. USA* **105** 18681–6
[21] Huang X, Schwenke D W, Tashkun S A and Lee T J 2012 An isotopic-independent highly accurate potential energy surface for CO_2 isotopologues and an initial $^{12}C^{16}O_2$ infrared line list *J. Chem. Phys.* **136** 124311
[22] Wheeler S E, Robertson K A, Allen W D, Schaefer H F III, Bomble Y J and Stanton J F 2007 Thermochemistry of key soot formation intermediates: C_3H_3 isomers *J. Phys. Chem.* A **111** 3819
[23] Buluta I and Nori F 2009 Quantum simulators *Science* **326** 108–11
[24] Barreiro J T, Müller M, Schindler P, Nigg D, Monz T, Chwalla M, Hennrich M, Roos C F, Zoller P and Blatt R 2011 An open-system quantum simulator with trapped ions *Nature* **470** 486–91
[25] Simon J, Bakr W S, Ma R, Tai M E, Preiss P M and Greiner M 2011 Quantum simulation of antiferromagnetic spin chains in an optical lattice *Nature* **472** 307–12
[26] Ma X-s, Dakic B, Naylor W, Zeilinger A and Walther P 2011 Quantum simulation of the wavefunction to probe frustrated Heisenberg spin systems *Nature Phys.* **7** 399–405
[27] Brown K L, Munro W J and Kendon V M 2010 Using quantum computers for quantum simulation *Entropy* **12** 2268–307
[28] Lanyon B P 2011 Universal digital quantum simulation with trapped ions *Science* **334** 57–61
[29] Preskill J 1997 Fault-tolerant quantum computation arXiv:quant-ph/9712048
[30] Nielsen M A and Chuang I L 2000 *Quantum Computation and Quantum Information* 1st edn (Cambridge: Cambridge University Press)
[31] Devitt S J, Nemoto K and Munro W J 2009 Quantum error correction for beginners arXiv:0905.2794
[32] Dawson C M and Nielsen M A 2006 The Solovay–Kitaev algorithm *Quantum Inform. Comput.* **6** 81
[33] Fowler A G 2011 Constructing arbitrary Steane code single logical qubit fault-tolerant gates *Quantum Inform. Comput.* **11** 867–73
[34] Gottesman D and Chuang I L 1999 Demonstrating the viability of universal quantum computation using teleportation and single-qubit operations *Nature* **402** 390–3
[35] Zhou X, Leung D W and Chuang I L 2000 Methodology for quantum logic gate construction *Phys. Rev.* A **62** 052316
[36] Høyer P and Špalek R 2005 Quantum fan-out is powerful *Theory Comput.* **1** 81
[37] Abrams D S and Lloyd S 1997 Simulation of many-body fermi systems on a universal quantum computer *Phys. Rev. Lett.* **79** 2586–9

[38] Abrams D S and Lloyd S 1999 Quantum algorithm providing exponential speed increase for finding eigenvalues and eigenvectors *Phys. Rev. Lett.* **83** 5162–5
[39] Grover L and Rudolph T 2002 Creating superpositions that correspond to efficiently integrable probability distributions arXiv:quant-ph/0208112
[40] Mohseni M and Lidar D A 2006 Direct characterization of quantum dynamics *Phys. Rev. Lett.* **97** 170501
[41] Ben-Shlomo S and Kaldor U 1988 The open-shell coupled-cluster method in general model space: five states of LiH *J. Chem. Phys.* **89** 956
[42] Du J, Xu N, Peng X, Wang P, Wu S and Lu D 2010 NMR implementation of a molecular hydrogen quantum simulation with adiabatic state preparation *Phys. Rev. Lett.* **104** 030502
[43] Jones N C, Van Meter R, Fowler A G, McMahon P L, Kim J, Ladd T D and Yamamoto Y 2012 Layered architecture for quantum computing *Phys. Rev.* X **2** 031007
[44] Kitaev A Yu 1995 Quantum measurements and the Abelian stabilizer problem arXiv:quant-ph/9511026v1
[45] Cleve R, Ekert A, Macchiavello C and Mosca M 1998 Quantum algorithms revisited *Proc. R. Soc. Lond.* A **454** 339–54
[46] Kitaev A Yu, Shen A H and Vyalyi M N 2002 *Classical and Quantum Computation* 1st edn (Providence, RI: American Mathematical Society)
[47] Cleve R and Watrous J 2000 Fast parallel circuits for the quantum Fourier transform *Proc. 41st Annu. Symp. on Foundations of Computer Science (Redondo Beach, CA, 2000)* pp 526–36
[48] Weinstein Y S, Pravia M A, Fortunato E M, Lloyd S and Cory D G 2001 Implementation of the quantum Fourier transform *Phys. Rev. Lett.* **86** 1889–91
[49] Vedral V, Barenco A and Ekert A 1996 Quantum networks for elementary arithmetic operations *Phys. Rev.* A **54** 147–53
[50] Draper T G 2000 Addition on a quantum computer arXiv:quant-ph/0008033
[51] Van Meter R and Itoh K M 2005 Fast quantum modular exponentiation *Phys. Rev.* A **71** 052320
[52] Cuccaro S A, Draper T G, Kutin S A and Moulton D P 2004 A new quantum ripple-carry addition circuit arXiv:quant-ph/0410184
[53] Draper T G, Kutin S A, Rains E M and Svore K M 2006 A logarithmic-depth quantum carry-lookahead adder *Quantum Inform. Comput.* **6** 351–69
[54] Isailovic N, Whitney M, Patel Y and Kubiatowicz J 2008 Running a quantum circuit at the speed of data *ISCA'08: 35th Int. Symp. on Computer Architecture (Beijing, 2008)*
[55] Knill E 2005 Quantum computing with realistically noisy devices *Nature* **434** 39–44
[56] DiVincenzo D P and Aliferis P 2007 Effective fault-tolerant quantum computation with slow measurements *Phys. Rev. Lett.* **98** 020501
[57] Fowler A G, Stephens A M and Groszkowski P 2009 High-threshold universal quantum computation on the surface code *Phys. Rev.* A **80** 052312
[58] Fowler A G and Hollenberg L C L 2004 Scalability of Shor's algorithm with a limited set of rotation gates *Phys. Rev.* A **70** 032329
Fowler A G and Hollenberg L C L 2007 Scalability of Shor's algorithm with a limited set of rotation gates *Phys. Rev.* A **75** 029905 (erratum)
[59] Helgaker T, Jorgensen P and Olsen J 2000 *Molecular Electronic-Structure Theory* (New York: Wiley)
[60] Schmidt M W *et al* 1993 General atomic and molecular electronic structure system *J. Comput. Chem.* **14** 1347–63
[61] Gordon M S and Schmidt M W 2005 Advances in electronic structure theory: GAMESS a decade later *Theory and Applications of Computational Chemistry: The First Forty Years* ed C E Dykstra, G Frenking, K S Kim and G E Scuseria (Amsterdam: Elsevier) pp 1167–89
[62] Hehre W J, Stewart R F and Pople J A 1969 Self-consistent molecular-orbital methods. I. Use of Gaussian expansions of Slater-type atomic orbitals *J. Chem. Phys.* **51** 2657
[63] Dunning T H 1971 Gaussian basis functions for use in molecular calculations. III. Contraction of (10s6p) atomic basis sets for the first-row atoms *J. Chem. Phys.* **55** 716

[64] Ward N J, Kassal I and Aspuru-Guzik A 2009 Preparation of many-body states for quantum simulation *J. Chem. Phys.* **130** 194105
[65] Suzuki M 1992 General theory of higher-order decomposition of exponential operators and symplectic integrators *Phys. Lett.* A **165** 387–95
[66] Wiebe N, Berry D, Høyer P and Sanders B C 2010 Higher order decompositions of ordered operator exponentials *J. Phys. A: Math. Theor.* **43** 065203
[67] Barenco A, Ekert A, Suominen K-A and Törmä P 1996 Approximate quantum Fourier transform and decoherence *Phys. Rev.* A **54** 139–46
[68] Van Meter R, Ladd T D, Fowler A G and Yamamoto Y 2010 Distributed quantum computation architecture using semiconductor nanophotonics *Int. J. Quantum Inform.* **8** 295–323
[69] Bocharov A and Svore K M 2012 Resource-optimal single-qubit quantum circuits *Phys. Rev. Lett.* **109** 190501
[70] Boghosian B M and Taylor W 1998 Simulating quantum mechanics on a quantum computer *Physica* D **120** 30–42
[71] Byrnes T and Yamamoto Y 2006 Simulating lattice gauge theories on a quantum computer *Phys. Rev.* A **73** 022328
[72] Lévi B, Georgeot B and Shepelyansky D L 2003 Quantum computing of quantum chaos in the kicked rotator model *Phys. Rev.* E **67** 046220